序

1984 年美国的 MathWorks 公司推出 Matlab, 到目前为止, 它已发展成为国际上最优秀的科技应用软件之一。其强大的科学计算与可视化功能、简单易用的开放式可扩展环境以及多达 30 多个面向不同领域而扩展的工具箱(Toolbox) 支持, 使得 Matlab 在许多学科领域中成为计算机辅助设计与分析、算法研究和应用开发的基本工具和首选平台。在我国, Matlab 已经拥有许多用户, 许多高校陆续开设了有关 Matlab 的课程, 清华大学、华中理工大学等高校的 BBS 上还专门设立了 Matlab 讨论区。

Matlab 最初用于自动控制系统的辅助设计, 而后采用了开放性开发的思想, 不断吸收各学科领域权威人士所编写的实用程序, 形成了一系列规模庞大、覆盖面极广的工具箱(Toolbox) 。所谓工具箱, 其实是一组一组的函数, 包括了通信系统、信号处理、图像处理、小波分析、鲁棒控制、系统辨识、非线性控制、模糊控制、神经网络、优化理论、样条、商用统计分析等等大量现代工程技术学科的内容, 非常实用。至今, 国内有关 Matlab 的书都把内容集中在 Matlab 语言和经典控制系统的设计上, 很少涉及 Matlab 工具箱, 致使大多数 Matlab 用户难以使用工具箱中丰富的函数。

在长期的学习、应用过程中, 我们感觉 Matlab 脚本式的语言其实不难掌握, 难点在于如何理解和掌握 Matlab 及其工具箱中大量函数的功能及用法, 避免重复性的劳动, 尽快地“站在巨人肩上”开展工作。本套书的目的就是希望通过详尽的介绍众多工具箱中的函数功能及使用方法, 以帮助用户更好更快地理解、掌握和使用 Matlab 及其工具箱。希望这套书能给 Matlab 用户们一点帮助, 让大家在学习和使用过程中少花点力气, 少走点弯路。

根据各个工具箱涉及的学科领域, 该套丛书分为 3 篇, 即信息工程篇、应用数学篇和控制工程篇, 共涉及 17 个工具箱。

信息工程篇包括信号处理工具箱(Signal Processing Toolbox) 、图像处理工具箱(Image Processing Toolbox) 、通信工具箱(Communications Toolbox) 、定点运算工具箱(Fixed - Point Blockset) 、小波分析工具箱(Wavelet Toolbox) 、高阶谱分析工具箱(High - Order Spectral Analysis Toolbox) 及地理信息处理工具箱(Mapping ToolBox) 。

应用数学篇包括统计工具箱(Statistics Toolbox) 、偏微分方程工具箱(Partial Differential Equation Toolbox) 、样条工具箱(Spline Toolbox) 及优化工具箱(Optimization Toolbox) 。

控制工程篇包括系统辨识工具箱(System Identification Toolbox) 、控制系统工具箱(Control System Toolbox) 、鲁棒控制工具箱(Robust Control Toolbox) 、模型预测控制工具箱(Model Predictive Control Toolbox) 、模糊逻辑工具箱(Fuzzy Logic Toolbox) 及非线性控制设计模块(Nonlinear Control Design Blockset) 。

本套丛书在简要阐明函数原理和算法的基础上, 给出了详细的函数使用说明, 大部分函数给出了应用实例。对于理工科学生和广大科技人员具有重要的参考价值。同时为完整起见, 以附录形式给出了 Matlab 标准环境下的所有函数的简单参考, 以供用户快速查询。

Matlab 工具箱涉及多个学科的理论知识, 内容非常丰富。由于我们水平和时间有限, 对于书中出现的错误和不妥之处, 恳请读者指正。

编著者

1999 年 11 月

前　　言

信息时代是应用数学大发展的时代。数学思想、数学方法与数学模型，随着计算机的广泛应用日益渗透到各个行业中。尤其是一些新兴的数学理论(统计学，运筹学，优化理论，有限元方法、模糊数学、样条等)已成为社会生产、科学实验、工程技术及经济管理中不可缺少的工具。

本书讲述 Matlab 中属于应用数学范畴的 4 个工具箱。它们是统计工具箱(Statistics Toolbox Ver 2.1)、偏微分方程工具箱(Partial Differential Equation Toolbox Ver 1.0)、样条工具箱(Spline Toolbox Ver 2.0)及优化工具箱(Optimization Toolbox Ver 5.0)。

统计工具箱能够支持范围广泛的统计计算任务，提供强大的工程和科学统计能力。其主要内容包括：多达 20 多种的概率分布、参数估计与假设检验、线性模型与非线性模型分析、多元统计分析、试验设计以及统计工序管理等。它不仅提供了大量的概率和统计的数值计算函数，同时提供了许多交互式图形工具函数以方便用户分析设计。

偏微分方程工具箱主要用于以有限元法求解偏微分方程的数值近似解，它可用于求解线性的椭圆型、双曲型及抛物线型偏微分方程，以及本征型方程和简单的非线性偏微分方程。在提供大量的计算函数以外，其强大的可视化能力将大大提高用户的分析能力。

样条工具箱提供最常用 B 形式和 PP 形式的大量样条计算函数，是学习样条和使用样条函数理想的软件环境。

优化工具箱的内容涵盖线性与非线性规划、二次规划、多目标决策、最小最大问题、半无限问题以及最小二乘与非线性方程求解等，并支持对多种优化算法的选择。

本书最后还附有 Matlab 的基本函数列表，供读者查询。

本书第一章由贺勇军执笔编写，第二章由张青斌执笔编写，第三章由刘志俭执笔编写，第四章和附录由李涛执笔编写。全书由李涛统稿，并由李涛、伯晓晨、徐昕审校。

在本书的成稿过程中，电子工业出版社的几位编辑在文稿的审阅、修订方面给我们提出了许多宝贵的意见，在此表示衷心的感谢。感谢父母和老师对我多年来的培养和教育。感谢刘朝晖小姐，她的关怀与帮助给我很大的精神鼓舞。感谢所有关心本书成长的朋友们。

李　涛

1999 年 11 月

目　　录

第1章　统计工具箱

(Statistics Toolbox Ver 2.1)

统计学是数据处理的科学和艺术，旨在通过收集、分析、解释和表达数据来探求事物中所蕴含的规律，达到发现新知识的目的。随着当今世界科学技术水平的迅猛发展，知识经济时代日益临近，人们需要处理越来越多的数据。因此，统计工作在国民经济和科学研究中的应用越来越广泛深入。Matlab 的统计工具箱则为人们提供了一个强有力的统计分析工具。

1.1　统计工具箱简介

统计工具箱是一套建立于 Matlab 数值计算环境的统计分析工具，能够支持范围广泛的统计计算任务，提供工程和科学统计的基本能力。其中包括 200 多个 M 文件（函数），主要支持以下各方面的内容。

- 概率分布——提供了 20 种概率分布类型，其中包括连续分布和离散分布，而且每种分布类型均给出 5 个有用的函数，即概率密度函数、累积分布函数、逆累积分布函数、随机数产生器和均值与方差计算函数。
- 参数估计——依据特定分布的原始数据，可以计算分布参数的估计值及其置信区间。
- 描述性统计——提供描述数据样本特征的函数，包括位置和散布的度量、分位数估计和处理数据缺失情况的函数等。
- 线性模型——针对线性模型，工具箱提供的函数涉及单因素方差分析、双因素方差分析、多重线性回归、逐步回归、响应曲面预测和岭回归等。
- 非线性模型——为非线性模型提供的函数涉及参数估计、多维非线性拟合的交互预测和可视化以及参数和预计值的置信区间计算等。
- 假设检验——此间提供最通用的假设检验的函数：t 检验和 z 检验。
- 多元统计——关于多元统计的函数有主成分分析和线性判别分析。
- 统计绘图——Matlab 图形库中添加了 box 图、正态概率图、威布尔概率图、管理图、分位数与分位数图等，另外还对多项式拟合和预测的支持进行扩展。
- 统计工序管理——可绘制通用的管理图和进行工序性能的研究。
- 试验设计——支持因子设计和 D 优化设计。

统计工具箱的函数主要分为两类：

- 数值计算函数

- 交互式图形工具函数

前一类工具由一些函数组成，可以通过命令行或自己的应用程序来调用这些函数。其中很多函数为 Matlab 的 M 文件，这些文件由一系列实现特殊统计算法的语句构成。可使用下述语句查看这些函数的代码：

```
type    function_name
```

也可以将 M 文件拷贝下来，然后进行修改，形成按您所需要的算法进行计算的 M 文件，并将其添加到工具箱中。

工具箱所提供的后一类工具是一些能够通过图形用户界面（GUI）来使用的交互式图形工具。这些基于 GUI 的工具同时也为多项式拟合和预测以及概率函数开发提供环境。

文中的函数参考或详解中包含各类函数使用的具体信息。对函数的描述包括函数调用格式、参数选项以及操作符的完整说明。许多函数说明中包括示例、函数算法的说明以及附加阅读材料的参考等等。

另外，统计工具箱中的函数所采用的数学符号符合以下惯例：

β　　线性模型中的参数

E（x）　　x 的期望值，$E(x)=\int tf(t)dt$

$f(x|a,b)$　　概率密度函数(x 为独立变量，a、b 为固定参数)

$F(x|a,b)$　　累积分布函数

$I([a,b])$　　指示（Indicator）函数

P 和 q　　p 为事件发生的概率，

　　q 为事件不发生的概率，故 $q=1-p$

1.2 概率分布及函数总览

随机变量的统计行为完全决定于其概率分布。随机变量的概率分布可以分为连续型、离散型和奇异型三大类。而实际中遇到的随机变量都是离散型、连续型或离散-连续型的，奇异型随机变量一般只在理论研究中出现。

统计工具箱中提供的概率分布为连续分布或离散分布，共 20 种，见表 1.2.1。

表 1.2.1 概率分布分类表

数据连续分布 (Continuous)	统计量连续分布 (Continuous)	离散分布 (Discrete)
Beta (β 分布)	Chi-square (x^2 分布)	Binomial(二项分布)
Exponential (指数分布)	Noncentral Chi-square (非中心 x^2 分布)	Discrete Uniform (离散均匀分布)
Gamma(γ 分布)	F (F 分布)	Geometric(几何分布)
Lognormal (对数正态分布)	Noncentral F (非中心 F 分布)	Hypergeometric (超几何分布)
Normal (正态(高斯)分布)	Student's t (学生 t 分布)	Negative Binomial (负二项分布)
Rayleigh(瑞利分布)	Noncentral t (非中心 t 分布)	Poisson(泊松分布)
Uniform(均匀分布)	——	——
Weibull(威布尔分布)	——	——

Matlab 为每种概率分布提供五类函数：

· 概率密度函数（pdf）
· 累积分布函数（cdf）
· 逆累积分布函数
· 随机数产生器
· 均值和方差

1.2.1　概率密度函数

对于离散分布和连续分布，其相应的概率密度函数 pdf（Probability Density Function）有各自不同的含义。

- 离散型随机变量：它是只有有穷个或可数个可能值的随机变量，其概率密度函数是观察到某一特定值的概率。
- 连续型随机变量：如果存在一非负函数 $p(x) \geq 0$，使对于任意实数 $a \leq b$，X 在区间(a,b)上的取值的概率为

  $$P\{a < X > b\} = \int_a^b p(x)\,dx$$

  则函数 $p(x)$ 称作 X 的概率密度函数，它满足

  $$\int_{-\infty}^{\infty} p(x)dx = 1$$

  与离散分布的 pdf 不同，其观察到某一特定值的概率为零。

pdf 具有两种性质：

· 对于每个可能的结果 pdf 为零或一正数；

· pdf 对整个区间的积分为 1。

pdf 并非单一函数，而是由一个或多个参数所表征的函数族。一旦选定（或估计）了参数值，此函数才唯一确定。

在统计工具箱中，对每种分布的 pdf 函数进行调用的格式是统一的，具体调用格式参见表 1.2.2。

下面以正态分布为例，说明 pdf 函数调用方法。

举例：

```
x=[-3:0.5:3];
f = normpdf(x,0,1);
f =
Columns 1 through 7
       0.0044    0.0175    0.0540    0.1295    0.2420
       0.3521    0.3989
Columns 8 through 13
    0.3521    0.2420    0.1295    0.0540    0.0175    0.0044
```

pdf 函数中的第一个参数提供所要计算其概率密度的点集(自变量 x)；其他参数提供能够

唯一确定分布的参数值，正态分布需要两个参数：位置参数（均值μ）和散度参数(标准差σ)。上例中，计算结果变量f则包含了由参数 0 和 1($\mu=0$，$\sigma=1$)所确定的正态分布函数在x取值上的概率密度。

在函数调用时，其中的参数可能是标量(即数量)、矢量或矩阵，因此在给定参数时，需要注意这些参量的长度(或称尺寸、大小等)应该相匹配。例如，β分布的 pdf 函数调用：P＝betacdf(X,A,B)。其中，X、A 和 B 的长度要么相同(如，它们都是单个标量，或都为包含 N 个元素的矢量或 N×M 个元素的矩阵)；要么，其中有的参数(假设为 X)是单个标量，而其他参量为矢量或矩阵，则 Matlab 自动将 X 扩展为与其他参量相同长度的矢量或矩阵，此矢量或矩阵的元素均为常量 X 的值。我们称这种自动操作方式为矢量扩展规则。

举例：

```
a=[0.5,0.5];
b=[0.5,1];
c=[0.5,1];
y=betapdf(a,b,c)
y =
    0.6366    1.0000
a = [0.5  1;  2  4]
a =
    0.5000 1.0000
    2.0000 4.0000
y = betapdf(0.5，a，a)
y =
    0.6366 1.0000
    1.5000 2.1875
```

在其他类似函数中，也通常采用矢量扩展规则对各参量进行操作。以后不再一一赘述。

除了表 1.2.2 中列出的特定分布的pdf函数外，统计工具箱还给出了通用的pdf调用函数，函数名即为 pdf。

- pdf

功能：可选分布的通用概率密度函数。

格式：Y = pdf('name',X,A1,A2,A3)

说明：Y = pdf('name',X,A1,A2,A3)提供了求取统计工具箱中任一分布的概率密度值功能。其中，‘name’为特定分布的名称，如‘Normal’、‘Gamma’等。X 为分布函数的自变量 x 的取值矩阵，而 A1、A2、A3 分别为相应的分布参数值。注意：由于各种分布所含参数不同，A1、A2、A3 的含义各不相同，也并不一定都是必须的；对于任一分布，A1、A2、A3 的值具体如何给出，可参见相应分布的特定概率密度函数。Y 存放结果，为概率密度值矩阵。

举例：
```
p = pdf('Normal',－2:2,0,1)
p =
```

```
    0.0540 0.2420 0.3989 0.2420 0.0540
p = pdf('Poisson',0:4,1:5)
p =
    0.3679 0.2707 0.2240 0.1954 0.1755
```

1.2.2　累积分布函数与逆累积分布函数

连续型随机变量的累积分布函数 cdf (Cumulative Distribution Function)，亦称分布函数，完全取决于其概率密度 $p(x)$，数学表达式为

$$F(x)=\int_{\infty}^{x} p(u)\,du$$

如果 f 是概率密度函数，则相应的累积分布函数（cdf）F 为

$$F(x)=P(X\le x)=\int_{\infty}^{x} f(t)dt \qquad (1.2.2)$$

累积分布函数 $F(x)$表示所观察结果小于或等于 x 的概率。cdf 具有两种性质：

• cdf 值 $F(x)$的范围为 0～1；

• 如果 $y\ge x$,则 $F(y)\ge F(x)$。

逆累积分布函数 icdf（Inverse Cumulative Distribution Function）返回给定显著概率条件下假设检验的临界值，实际上是 cdf 的逆函数。

在统计工具箱中，对每种分布的 cdf 和 icdf 函数(名称以 inv 结尾)进行调用的格式是统一的，参见表 1.2.3。

另外，工具箱提供了通用的累积分布函数 cdf 和逆累积分布函数 icdf，说明如下。

• cdf, icdf

功能：计算可选分布的累积分布函数和逆累积分布函数。

格式：P = cdf('name',X,A1,A2,A3)

　　　X = icdf('name',P,A1,A2,A3

说明：P = cdf('name',X,A1,A2,A3)与 pdf 函数的区别仅在于它是计算某种分布的累积分布函数值，而不是概率密度值，其他用法与 pdf 函数相同。

　　　X = icdf('name',P,A1,A2,A3)为 P = cdf('name',X,A1,A2,A3)的逆函数。

举例：

```
p = cdf('Normal',－2:2,0,1)
p =
      0.0228 0.1587 0.5000 0.8413 0.9772
p = cdf('Poisson',0:5,1:6)
p =
      0.3679 0.4060 0.4232 0.4335 0.4405 0.4457
x = icdf('Normal',0.1:0.2:0.9,0,1)
x =
```

```
        －1.2816  －0.5244   0   0.5244   1.2816
    x = icdf('Poisson',0.1:0.2:0.9,1:5)
    x =
          1 1 3 5 8
```

参见：mle, pdf, random。

下面说明正态分布的 cdf 函数调用方法。

```
x = [－3:0.1:3];
p = normcdf(x,0,1);
```

其中，变量 p 包含由参数 0 和 1 所确定的正态分布函数在 x 中所取值上的累积分布函数值。所用参数含义与 pdf 函数类同。

下面说明连续的累积分布函数(cdf)与其逆函数(icdf)的关系。

```
x = [－3:0.1:3];
xnew = norminv(normcdf(x,0,1),0,1);
```

相反地，进行下述计算：

```
p = [0.1:0.1:0.9];
pnew = normcdf(norminv(p,0,1),0,1);
```

请对照一下 x 与 xnew 和 p 与 pnew，可以发现其中的规律。

连续分布中取值点的 cdf 计算值为 0～1 的概率值，这些概率值的逆 cdf 则给出其原来的取值点。

对于离散分布，cdf 与其 icdf 的关系更为复杂些。因为很可能不存在某个值(设为 x)，使得 x 的 cdf 为 p。在这种情况下，其 icdf 返回使 cdf(x)⩾p 的第一个值 x′。如：

```
x = [0:10];
y = binoinv(binocdf(x,10,0.5),10,0.5);
```

请对照一下 x 与 y。

以下的命令说明了进行相反操作所同样存在的问题。

```
p = [0.1:0.2:0.9];
pnew = binocdf(binoinv(p,10,0.5),10,0.5)
pnew =
    0.1719 0.3770 0.6230 0.8281 0.9453
```

逆函数在假设检验和产生置信区间等工作中是很有用的。以下给出获得正态分布的 99％置信区间的方法。

```
p = [0.005 0.995];
x = norminv(p,0,1)
x =
    －2.5758 2.5758
```

变量 x 中的值即为给定概率区间 p 的条件下，由参数 0 和 1 所确定的正态分布函数的逆函数的结果，p(2)－p(1)=0.99。因此，x 给出了标准正态分布的 99％置信区间。

统计工具箱给出统一的调用每种分布的逆函数的格式。第一个输入参数提供概率数集，

其他参数给出确定的分布参数值。

1.2.3　随机数产生器

所有分布的随机数的产生方法都始于均匀分布随机数。一旦具备了均匀分布随机数产生器，其他分布的随机数都可使用直接法、反转 (inversion)法或拒绝(rejection)法获得。

(1)　随机数产生的基本方法

1　直接法(direct)

直接法源于分布的定义。假设产生二项分布随机数，可以认为其随机数就是在 n 次抛硬币之后，某一面出现的次数(每次抛掷时，此面出现的概率为 p)。如果产生了 n 个均匀分布随机数，数出大于 p 的次数 m，那么结果 m 就是参数为 n 和 p 的二项分布的随机数。

2　反演法 (inversion)

反演方法的理论基础在于均匀分布与其他连续分布之间的关系。假设 F 为一连续分布，其逆为 F^{-1}；U 是一个均匀分布随机数，则 $F^{-1}(U)$服从 F 分布。因此，可将某种分布的逆函数作用于均匀分布的随机数，获得这种分布的随机数。遗憾的是，这种方法通常并不是最有效的。

3　拒绝法(rejection)

对于某些分布，其函数形式使得运用直接法和反演法来产生随机数比较困难或费时较多。在这种情况下，拒绝法或许能很好地解决这一问题。

若要产生概率密度函数为 f 的某种分布的随机数，若采用拒绝法，首先需找到另一分布密度函数 g 和一个常数 c，并满足以下条件：

$$f(x) \le cg(x)\ ,\forall x \tag{1.2.3}$$

然后通过以下步骤进行：

1. 产生概率密度为 $g(x)$ 的 G 分布的随机数 x；
2. 设参数 $r=\dfrac{cg(x)}{f(x)}$；
3. 产生一个均匀分布随机数 u；
4. 如果 $r\times u<1$,返回 x；
5. 否则重复第 1 步至第 3 步。

为提高效率，产生 G 分布随机数的方法要简单，而且常数 c 的值要小。迭代的期望值为 c。

(2) 产生随机数的通用函数

在工具箱中，提供了通用的随机数产生函数random和特定分布的随机数产生函数(以rnd结尾)。可以直接调用这些函数来获得所需的随机数，而不必经过(1)中所述的过程。

• random

功能：产生可选分布的随机数。

格式：y = random('name',A1,A2,A3,m,n)

说明：random 函数产生统计工具箱中任一分布的随机数。'name'为相应分布的名称。A1、A2 和 A3 为分布参数，其意义同 pdf 函数中的说明。参数 m、n 确定了结果 y 的数量，如果分布参数 A1、A2 和 A3 为标量，则 y 以 m×n 矩阵形式给出；如果 A1、A2 和 A3 为矢量，则 m,n 是可选的，但应注意，它们所给出的长度或矩阵行列数必须与分布参数的长度相匹配。

举例：
```
rn = random('Normal',0,1,2,4)
rn =
1.1650 0.0751 -0.6965 0.0591
0.6268 0.3516 1.6961 1.7971
rp = random('Poisson',1:6,1,6)
rp =
0  0  1  2  5  7
```

(3) 特定分布的随机数产生函数

对于 20 种分布类型，每种分布的随机数都可任意产生。函数可以产生单个随机数或随机数矩阵，这取决于函数调用时所采用的参数。所有特定分布的随机数产生函数列在表 1.2.4 内。下面说明产生β分布随机数的方法。四条语句分别给出不同的随机数。

```
a = 1;
b = 2;
c = [.1 .5; 1 2];
d = [.25 .75; 5 10];
m = [2 3];
nrow = 2;
ncol = 3;
r1 = betarnd(a,b)
r1 =
   0.4469
r2 = betarnd(c,d)
r2 =
   0.8931 0.4832
   0.1316 0.2403
```

```
r3 = betarnd(a,b,m)
r3 =
   0.4196 0.6078 0.1392
   0.0410 0.0723 0.0782
r4 = betarnd(a,b,nrow,ncol)
r4 =
   0.0520 0.3975 0.1284
   0.3891 0.1848 0.5186
```

1.2.4　均值和方差

概率分布的均值和方差是分布参数的简单函数。在统计工具箱中，用‘stat’结尾的函数可以计算得到给定参数的某种分布的均值和方差。所有计算均值和方差的函数见表 1.2.5。下面绘制给定参数后威布尔分布的均值的曲线图(图 1.2.1)。

```
x = (0.5:0.1:5);
y = (1:0.04:2);
[X,Y] = meshgrid(x,y);
Z = weibstat(X,Y);
[c,h] = contour(x,y,Z,[0.4 0.6 1.0 1.8]);
clabel(c);
```

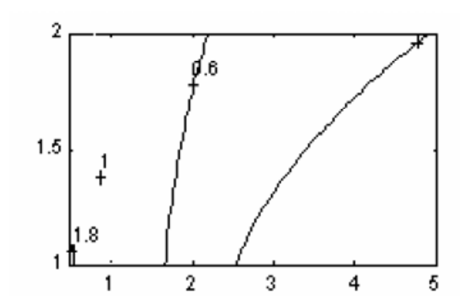


图 1.2.1　威布尔分布的均值曲线图

表 1.2.2　概率密度函数(pdf)

分布类型名称	概率密度函数数学定义	函数名称	调 用 格 式	备　注
β 分布	$y=f(x\mid a,b)=\dfrac{1}{B(a,b)}x^{a-1}(1-x)^{b-1}I_{(0,1)}(x)$ $(0<x<1)$	betapdf	P = betapdf(X,A,B)	a>0,b>0,x 在 [0,1] 区间内取值
二项分布	$y=f(x\mid n,p)=\begin{pmatrix}n\\x\end{pmatrix}p^{x}q^{(1-x)}I_{(0,1,\ldots,n)}(x)$ 其中，$\begin{pmatrix}n\\x\end{pmatrix}=\dfrac{n!}{x!(n-x)!}$ ，$q=1-p$	binopdf	Y = binopdf(X,N,P)	N 为正整数， P 在[0,1]区间内取值
χ^2 分布	$y=f(x\mid v)=\dfrac{x^{(v-2)/2}e^{-x/2}}{2^{v/2}\Gamma(v/2)}$	chi2pdf	Y = chi2pdf(X,V)	自由度 V 为正整数
指数分布	$y=f(x\mid\mu)=\dfrac{1}{\mu}e^{-\frac{x}{\mu}}$	exppdf	Y = exppdf(X,MU)	参数 MU 为正整数
F 分布	$y=f(x\mid v_1,v_2)=\dfrac{\Gamma\left[\frac{v_1+v_2}{2}\right]}{\Gamma\left(\frac{v_1}{2}\right)\Gamma\left(\frac{v_2}{2}\right)}\left(\dfrac{v_1}{v_2}\right)^{\frac{v_1}{2}}\dfrac{x^{\frac{v_1-2}{2}}}{\left[1+\left(\frac{v_1}{v_2}\right)x\right]^{\frac{v_1+v_2}{2}}}$	fpdf	Y = fpdf(X,V1,V2)	参数V1和V2为正整数。X 在区间[0, ∞]内取值
γ 分布	$y=f(x\mid a,b)=\dfrac{1}{b^{a}\Gamma(a)}x^{a-1}e^{-\frac{x}{b}}$	gampdf	Y = gampdf(X,A,B)	参数 A 和 B 为正整数。X 在区间[0, ∞]内取值
几何分布	$y=f(x\mid p)=pq^{x}\mathrm{I}_{(0,1,K)}(x)$ ，其 中 ，$q=1-p$	geopdf	Y = geopdf(X,P)	参数 P 在区间[0,1]内取值
超几何分布	$y=f(x\mid M,K,n)=\dfrac{\begin{pmatrix}K\\x\end{pmatrix}\begin{pmatrix}M-K\\n-x\end{pmatrix}}{\begin{pmatrix}M\\n\end{pmatrix}}$	hygepdf	Y = hygepdf(X,M,K,N)	参数 M、K、N 为正整数；参数 X 小于或等于其他参数值；参数 N 小于或等于 M
正态分布	$y=f(x\mid\mu,\sigma)=\dfrac{1}{\sigma\sqrt{2\pi}}e^{\frac{-(x-\mu)^2}{2\sigma^2}}$	normpdf	Y = normpdf(X,MU, SIGMA)	参数 SIGMA 为正数
对数正态分布	$y=f(x\mid\mu,\sigma)=\dfrac{1}{x\sigma\sqrt{2\pi}}e^{\frac{-(\ln x-\mu)^2}{2\sigma^2}}$	lognpdf	Y = lognpdf(X,MU, SIGMA)	
负二项分布	$y=f(x\mid r,p)=\begin{pmatrix}r+x+1\\x\end{pmatrix}p^{r}q^{x}\mathrm{I}_{(0,1,\ldots)}(x)$ 其中，$q=1-p$	nbinpdf	Y = nbinpdf(X,R,P)	
非中心 F 分布	设随机变量 $\chi_1^2(\mu)$ 有自由度为 f_1 和非中心参数为 μ 的非中心 χ^2 分布， χ_2^2 有自由度为 f_2 的 χ^2 分布，并且 $\chi_1^2(\mu)$ 和 χ_2^2 独立，则随机变量 $F=\dfrac{\chi_1^2(\mu)/f_1}{\chi_2^2/f_2}$ 的分布，称作自由度为(f_1 , f_2)和非中心参数为 μ 的非中心分布	ncfpdf	Y = ncfpdf(X,NU1,NU2, DELTA)	分子自由度（df）为 NU1、分母自由度（df）为 NU2、正的非中心参数为 DELTA

(续表)

分布类型名称	概率密度函数数学定义	函数名称	调 用 格 式	备　　注
非中心 t 分布	称随机变量 $t(\mu)=\frac{U+\mu}{\sqrt{\chi_{(v)}^2/v}}$ 的分布为自由度为 v 和非中心参数为 μ 的非中心 t 分布，如果 1）U~N(μ ,1)；2） $\chi_{(v)}^2$ 服从自由度为 v 的 χ^2 分布；3)U 与 $\chi_{(v)}^2$ 独立。	nctpdf	Y = nctpdf(X,V,DELTA)	自由度（df）为 V、非中心参数为 DELTA
非中心 χ^2 分布	若随机变量 $X_i \sim N(\mu_i,\sigma^2)(i=1,\Lambda,\nu)$ 相互独立，则随机变量 $\chi^2(\mu)=(X_1^2+\Lambda+X_\nu^2)/\sigma^2$ 的分布，称作自由度为 v 的非中心 χ^2 分布，而 $\mu^2=(\mu_1^2+\Lambda+\mu_\nu^2)/\sigma^2$ 称作分布的非中心参数	ncx2pdf	Y = ncx2pdf(X,V,DELTA)	自由度（df）为 V、正非中心参数为 DELTA
泊松分布	$y=f(x\|\lambda)=\frac{\lambda^x}{x!}e^{-\lambda}\mathrm{I}_{(0,1,K)}(x)$	poisspdf	Y = poisspdf(X,LAMBDA)	参数 LAMBDA（λ）必须为正数。参数 X 可为任何非负整数。除非 X 为整数，否则概率密度函数值为 0
瑞利分布	$y=f(x\|b)=\frac{x}{b^2}e^{\left(\frac{-x^2}{2b^2}\right)}$	raylpdf	Y = raylpdf(X,B)	
学生 t 分布	$y=f(x\|\nu)=\frac{\Gamma\left(\frac{\nu+1}{2}\right)}{\Gamma\left(\frac{\nu}{2}\right)}\frac{1}{\sqrt{\nu\pi}}\frac{1}{\left(1+\frac{x^2}{\nu}\right)^{\frac{\nu+1}{2}}}$	tpdf	Y = tpdf(X,V)	自由度 V 为正整数
离散均匀分布	$y=f(x\|N)=\frac{1}{N}I_{(1,\ldots,N)}(x)$	unidpdf	Y = unidpdf(X,N)	参数 N 为正整数
连续均匀分布	$y=f(x\|a,b)=\frac{1}{b-a}\mathrm{I}_{[a,b]}(x)$	unifpdf	Y = unifpdf(X,A,B)	A<B
威布尔分布	$y=f(x\|a,b)=abx^{b-1}e^{-ax^b}\mathrm{I}_{(0,\infty)}(x)$	weibpdf	Y = weibpdf(X,A,B)	参数 A、B 为正数。某些文献中，威布尔分布的概率密度函数仅含一个参数（B），此时，A＝1

表 1.2.3 累积分布函数(cdf)与逆累积分布函数（inv）

分布类型名称	累积分布函数数学定义	函数名称	函数调用格式	逆累积分布函数名称及调用格式
β 分布	$p=F(x\|a,b)=\frac{1}{B(a,b)\int_0^x t^{a-1}(1-t)^{b-1}dt}$	betacdf	P = betacdf(X,A,B)	X = betainv(P,A,B)
二项分布	$y=F(x\|n,p)=\sum_{i=0}^{x}\binom{n}{i}p^i q^{(1-i)} I_{(0,1,\ldots,n)}(i)$	binocdf	Y = binocdf(X,N,P)	X = binoinv(Y,N,P)
χ^2 分布	$p=F(x\|v)=\int_0^x \frac{t^{(v-2)/2}e^{-t/2}}{2^{v/2}\Gamma(v/2)}dt$	chi2cdf	P = chi2cdf(X,V)	X = chi2inv(P,V)
指数分布	$p=F(x\|\mu)=\int_0^x \frac{1}{\mu}e^{-\frac{x}{\mu}}dt=1-e^{-\frac{x}{\mu}}$	expcdf	P = expcdf(X,MU)	X = expinv(P,MU)
F 分布	$F(x\|\nu_1,\nu_2)=\int_0^x \frac{\Gamma\left[\frac{\nu_1+\nu_2}{2}\right]}{\Gamma\left(\frac{\nu_1}{2}\right)\Gamma\left(\frac{\nu_2}{2}\right)}\left(\frac{\nu_1}{\nu_2}\right)^{\frac{\nu_1}{2}}\frac{t^{\frac{\nu_1-2}{2}}}{\left[1+\left(\frac{\nu_1}{\nu_2}\right)t\right]^{\frac{\nu_1+\nu_2}{2}}}dt$	fcdf	P = fcdf(X,V1,V2)	X = finv(P,V1,V2)
γ 分布	$p=F(x\|a,b)=\frac{1}{b^a\Gamma(a)}\int_0^x t^{a-1}e^{-\frac{t}{b}}dt$	gamcdf	P = gamcdf(X,A,B)	X =gaminv(P,A,B)
几何分布	$y=F(x\|p)=\sum_{i=0}^{floor(x)} pq^i$ ，$q=1-p$	geocdf	Y = geocdf(X,P)	X = geoinv(Y,P)
超几何分布	$p=F(x\|M,K,N)=\sum_{i=0}^{x}\frac{\binom{K}{i}\binom{M-K}{N-i}}{\binom{M}{N}}$	hygecdf	P = hygecdf(X,M, K,N)	X = hygeinv(P,M, K,N)
正态分布	$p=f(x\|\mu,\sigma)=\frac{1}{\sigma\sqrt{2\pi}}\int_{-\infty}^{x} e^{\frac{-(t-\mu)^2}{2\sigma^2}}dt$	normcdf	P = normcdf(X,MU, SIGMA)	X = norminv(P,MU, SIGMA)
对数正态分布	$p=f(x\|\mu,\sigma)=\frac{1}{x\sigma\sqrt{2\pi}}\int_0^x \frac{e^{\frac{-(\ln t-\mu)^2}{2\sigma^2}}}{t}dt$	logncdf	P = logncdf(X,MU, SIGMA)	X= logninv(P,MU, SIGMA)
负二项分布	$y=F(x\|r,p)=\sum_{i=0}^{x}\binom{r+i+1}{i}p^r q^i \mathrm{I}_{(0,1,\ldots)}(i)$	nbincdf	Y = nbincdf(X,R,P)	X = nbininv(Y,R,P)
非中心 F 分布	$F(x\|\nu_1,\nu_2,\delta)=\sum_{j=0}^{\infty}\left(\frac{\left(\frac{1}{2}\delta\right)^j}{j!}e^{-\frac{\delta}{2}}\right)\mathrm{I}\left(\frac{\left(\frac{\nu_1 x}{\nu_2+\nu_1 x}\right)}{\nu_1/2+j},\nu_2/2\right)$ I（$x\|a,b$) 为参数为 a 和 b 的不完全 β 分布函数	ncfcdf	P=ncfcdf(X,NU1, NU2,DELTA)	X = ncfinv(P,NU1, NU2,DELTA)
非中心 t 分布	$\Pr((-t)<x<t\|\nu,\delta)=\sum_{j=0}^{\infty}\left(\frac{\left(\frac{1}{2}\delta^2\right)^j}{j!}e^{-\frac{\delta^2}{2}}\right)\mathrm{I}\left(\frac{\left(\frac{x^2}{\nu+x^2}\right)}{\nu/2+j},\nu/2\right)$	nctcdf	P = nctcdf(X,NU, DELTA)	X = nctinv(P,NU, DELTA)
非中心 χ^2 分布	$F(x\|v,\delta)=\sum_{j=0}^{\infty}\left(\frac{\left(\frac{1}{2}\delta\right)^j}{j!}e^{-\frac{\delta}{2}}\right)\Pr[\chi^2_{v+2j}\le x]$	ncx2cdf	P = ncx2cdf(X,V, DELTA)	X = ncx2inv(P, V,DELTA)

(续表)

分布类型名称	累积分布函数数学定义	函数名称	函数调用格式	逆累积分布函数名称及调用格式
泊松分布	$p=F(x\mid\lambda)=e^{-\lambda}\sum_{i=0}^{floor(x)}\frac{\lambda^i}{i!}$	poisscdf	P = poisscdf(X, LAMBDA)	X = poissinv(P, LAMBDA)
瑞利分布	$y=F(x\mid b)=\int_0^x \frac{t}{b^2}e^{\left(\frac{-t^2}{2b^2}\right)}dt$	raylcdf	P = raylcdf(X,B)	X = raylinv(P,B)
学生 t 分布	$p=F(x\mid\nu)=\int_{-\infty}^{x}\frac{\Gamma\left(\frac{\nu+1}{2}\right)}{\Gamma\left(\frac{\nu}{2}\right)}\frac{1}{\sqrt{\nu\pi}}\frac{1}{\left(1+\frac{t^2}{\nu}\right)^{\frac{\nu+1}{2}}}dt$	tcdf	P = tcdf(X,V)	X = tinv(P,V)
离散均匀分布	$p=F(x\mid N)=\frac{floor(x)}{N}I_{(1,\mathrm{K},N)}(x)$	unidcdf	P = unidcdf(X,N)	X = unidinv(P,N)
连续均匀分布	$p=F(x\mid a,b)=\frac{x-a}{b-a}\mathrm{I}_{[a,b]}(x)$	unifcdf	P = unifcdf(X,A,B)	X = unifinv(P,N)
威布尔分布	$p=F(x\mid a,b)=\int_0^x abt^{b-1}e^{-at^b}dt=1-e^{-ax^b}I_{(0,\infty)}(x)$	weibcdf	P = weibcdf(X,A,B)	X = weibinv(P,A,B)

表 1.2.4　统计工具箱的随机数产生函数

分布类型名称	函数名称	函数调用格式		
		格式 1	格式 2	格式 3
β 分布	betarnd	R = betarnd(A,B)	R = betarnd(A,B,m)	R = betarnd(A,B,m,n)
二项分布	binornd	R = binornd(N,P)	R = binornd(N,P,mm)	R = binornd(N,P,mm,nn)
χ^2 分布	chi2rnd	R = chi2rnd(V)	R = chi2rnd(V,m)	R = chi2rnd(V,m,n)
指数分布	exprnd	R = exprnd(MU)	R = exprnd(MU,m)	R = exprnd(MU,m,n)
F 分布	frnd	R = frnd(V1,V2)	R = frnd(V1,V2,m)	R = frnd(V1,V2,m,n)
γ 分布	gamrnd	R = gamrnd(A,B)	R=gamrnd(A,B,m)	R = gamrnd(A,B,m,n)
几何分布	geornd	R = geornd(P)	R = geornd(P,m)	R = geornd(P,m,n)
超几何分布	hygernd	R = hygernd(M,K,N)	R = hygernd(M,K,N,mm)	R = hygernd(M,K,N,mm,nn)
对数正态分布	lognrnd	R = lognrnd(MU,SIGMA)	R = lognrnd(MU,SIGMA,m)	R = lognrnd(MU,SIGMA,m,n)
负二项分布	nbinrnd	R = nbinrnd(R,P)	R = nbinrnd(R,P,m)	R = nbinrnd(R,P,m,n)
非中心 F 分布	ncfrnd	R = ncfrnd(NU1, NU2,DELTA)	R = ncfrnd(NU1,NU2, DELTA,m)	R = ncfrnd(NU1,NU2, DELTA,m,n)
非中心 t 分布	nctrnd	R = nctrnd(V,DELTA)	R = nctrnd(V,DELTA,m)	R = nctrnd(V,DELTA,m,n)
非中心 χ^2 分布	ncx2rnd	R = ncx2rnd(V,DELTA)	R = ncx2rnd(V,DELTA,m)	R = ncx2rnd(V,DELTA,m,n)
正态分布	normrnd	R = normrnd(MU,SIGMA)	R = normrnd(MU,SIGMA,m)	R = normrnd(MU,SIGMA,m,n)
泊松分布	poissrnd	R = poissrnd(LAMBDA)	R = poissrnd(LAMBDA,m)	R = poissrnd(LAMBDA,m,n)
瑞利分布	raylrnd	R = raylrnd(B)	R = raylrnd(B,m)	R = raylrnd(B,m,n)
学生 t 分布	trnd	R = trnd(V)	R = trnd(V,m)	R = trnd(V,m,n)
离散均匀分布	unidrnd	R = unidrnd(N)	R = unidrnd(N,mm)	R = unidrnd(N,mm,nn)
连续均匀分布	unifrnd	R = unifrnd(N)	R = unifrnd(N,mm)	R = unifrnd(N,mm,nn)
威布尔分布	weibrnd	R = weibrnd(A,B)	R = weibrnd(A,B,m)	R = weibrnd(A,B,m,n)

表 1.2.5 分布的均值和方差计算函数

分布类型名称	数学定义		函数名称及调用格式
	均值	方差	
β分布	$\frac{a}{a+b}$	$\frac{ab}{(a+b+1)(a+b)^2}$	[M,V] = betastat(A,B)
二项分布	np	npq，$q=1-p$	[M,V] = binostat(N,P)
χ^2 分布	n	$2n$	[M,V] = chi2stat(NU)
指数分布	μ	μ^2	[M,V] = expstat(MU)
F分布	$\frac{v_2}{v_2-2}$，$v_2>2$	$\frac{2{v_2}^2(v_1+v_2-2)}{v_1(v_2-2)^2(v_2-4)}$，$v_2>4$	[M,V] = fstat(V1,V2) 若 $v_2<3$，F 分布的均值无定义；若 $v_2<5$，F 分布的方差无定义
γ分布	ab	ab^2	[M,V] = gamstat(A,B)
几何分布	q/p	q/p^2，$q=1-p$	[M,V] = geostat(P)
超几何分布	$N\frac{K}{M}$	$N\frac{K}{M}\frac{M-K}{M}\frac{M-N}{M-1}$	[MN,V] = hygestat(M,K,N)
正态分布	μ	σ^2	[M,V] = normstat(MU, SIGMA)
对数正态分布	$e^{\left(\mu+\frac{\sigma^2}{2}\right)}$	$e^{\left(2\mu+2\sigma^2\right)}-e^{\left(2\mu+\sigma^2\right)}$	[M,V] = lognstat(MU,SIGMA)
负二项分布	$\frac{rq}{p}$	$\frac{rq}{p^2}$，$q=1-p$	[M,V] = nbinstat(R,P)
非中心F分布	$\frac{v_2(\delta+v_1)}{v_1(v_2-2)}$，$v_2>2$	$2\left(\frac{v_2}{v_1}\right)^2\left[\frac{(\delta+v_1)^2+(2\delta+v_1)(v_2-2)}{(v_2-2)^2(v_2-4)}\right]$ $v_2>4$	[M,V] = ncfstat(NU1, NU2,DELTA)
非中心t分布	$\frac{\delta\,(v/2)^{1/2}\Gamma((v-1)/2)}{\Gamma(v/2)}$	$\frac{v}{(v-2)}(1+\delta^2)-\frac{v}{2}\delta^2\left[\frac{\Gamma((v-1)/2)}{\Gamma(v/2)}\right]^2$	[M,V] = nctstat(NU,DELTA)
非中心 χ^2 分布	$v+\delta$	$2(v+2\delta)$	[M,V] = ncx2stat(NU,DELTA)
泊松分布	λ	λ	[M,V] = poisstat(LAMBDA)
瑞利分布	$b\left(\frac{\pi}{2}\right)^{\frac{1}{2}}$	$\frac{2-\pi}{2}b^2$	[M,V] = raylstat(B)
学生t分布	$v>1$ 时，均值为 0; $v=1$，均值不存在	$v>2$ 时，$\frac{v}{v-2}$	[M,V] = tstat(NU)
离散均匀分布	$\frac{N+1}{2}$	$\frac{N^2-1}{12}$	[M,V] = unidstat(N)
连续均匀分布	$(a+b)/2$	$(b-a)/12$	[M,V] = unidstat(A,B)
威布尔分布	$a^{-\frac{1}{b}}\Gamma(1-b^{-1})$	$a^{-\frac{2}{b}}\left[\Gamma(1+2b^{-1})-\Gamma^2(1+b^{-1})\right]$	[M,V] = weibstat(A,B)

1.3 参数估计

参数估计是总体分布的数学形式已知，且可以用有穷个参数表示的估计问题。估计问题可以分为点估计和区间估计两个子问题。在参数模型中，使用最为广泛的一种参数估计方法

就是极大似然法(Maximum likelihood estimation，简称 MLE)，它是统计学奠基人之一费希尔(R.A.Fisher)首先提出来的，具有非常优良的统计性质。

统计工具箱即采用极大似然法给出常用的概率分布模型的参数的点估计和区间估计值。除此之外，还提供了部分分布的对数似然函数的计算功能。

工具箱所提供的参数估计函数和对数似然函数见表 1.3.1。

表 1.3.1 参数估计函数

函 数 名 称	功 能
mle	最大似然估计
betafit	β分布的参数估计
betalike	β分布的负对数似然函数
binofit	二项分布的参数估计
expfit	指数分布的参数估计
gamfit	γ分布的参数估计
gamlike	γ分布的负对数似然函数
normlike	正态分布的负对数似然函数
normfit	正态分布的参数估计
poissfit	泊松分布的参数估计
unifit	均匀分布的参数估计
weibfit	威布尔分布的参数估计
weiblike	威布尔分布的负对数似然函数

1.3.1 参数估计函数

各参数估计函数的调用方法是基本相同的，在此以β分布的参数估计函数调用为例，说明其格式和用法。另外给出了正态分布和泊松分布运用的程序示例。

1 betafit

功能：β分布数据的参数点估计和置信区间。

格式：phat = betafit(x)

[phat,pci] = betafit(x,alpha)

说明：betafit 函数计算并返回来自数据样本 X 的β分布的参数 a 和 b 的最大似然估计 phat 及其置信区间 pci。置信区间结果 pci 以 2×2 矩阵的形式给出，其第一列为参数 a 的置信下界和上界，第二列为参数 b 的置信下界和上界。可选的输入参数 alpha 为置信度，可控制置信区间的宽度。缺省情况下，alpha 为 0.05，对应 95% 置信区间。若 alpha＝0.01，则对应 99% 置信区间。

举例：随机产生 100 个β分布数据，相应的分布参数真值为 4 和 3。注意，ci 的各列包括参数的真值。

```
r = betarnd(4,3,100,1);
[p,ci] = betafit(r,0.01)
p =
3.9010 2.6193
ci =
2.5244 1.7488
```

```
5.2777 3.4899
```

参考：Hahn, Gerald J., & Shapiro, Samuel, S. "Statistical Models in Engineering", Wiley Classics Library John Wiley & Sons, NewYork1994. p. 95。

参见：betalike, mle。

下面的两个例子可进一步予以参考。

例1

```
r = normrnd(10,2,100,2);
[mu,sigma,muci,sigmaci] = normfit(r)
mu =
   10.1455 10.0527
sigma =
    1.9072 2.1256
muci =
   9.7652 9.6288
   10.5258 10.4766
sigmaci =
   1.6745 1.8663
   2.2155 2.4693
```

例2　泊松分布的样本均值即为λ的最大似然估计。

$$\lambda=\frac{1}{n}\sum_{i=1}^{n}x_i \tag{1.3.1}$$

```
r = poissrnd(5,10,2);
[l,lci] = poissfit(r)
l =
   4.8000 4.8000
lci =
   3.5000 3.5000
   6.2000 6.2000
```

统计工具箱还提供了通用的最大似然估计函数mle，可以单独完成各种分布的参数估计。

2 mle

功能：求取分布参数的最大似然估计量。

格式：phat = mle('dist',data)

[phat,pci] = mle('dist',data)

[phat,pci] = mle('dist',data,alpha)

[phat,pci] = mle('dist',data,alpha,p1)

说明：‘dist’参量为用户所给出的特定分布的名称，如‘beta’(指β分布)、‘binomial’(指二项分布)等。data为数据样本，以行矢量形式给出。Alpha是用户给定的

置信度值，以提供函数给出相应的置信度为 100(1-Alpha)%的置信区间。如 Alpha=0.03，则函数给出置信度为 97%的置信区间。若不给出 Alpha，即在缺省情况下，如前两种格式，函数给出置信度为 95%的置信区间。而调用格式[phat,pci] = mle('dist',data,alpha,p1)是仅供二项分布的参数估计所使用，p1 为试验的次数。

举例：
```
rv = binornd(20,0.75)
rv =
    16
[p,pci] = mle('binomial',rv,0.05,20)
p =
    0.8000
pci =
    0.5634
    0.9427
```

参见：betafit, binofit, expfit, gamfit, normfit, poissfit, weibfit。

1.3.2 对数似然函数

统计工具箱提供了β分布、γ分布、正态分布和威布尔分布等分布类型的负对数似然函数值的求取函数，同时能够给出 Fisher 信息量，以供用户使用。以下分别进行详细说明。

1 betalike

功能：负β分布对数似然函数。

格式：logL = betalike(params,data)

[logL,info] = betalike(params,data)

说明：logL = betalike(params,data)返回β分布对数似然函数的负值。其中 params 为包含β分布的参数 a、b 的矢量[a,b]，data 为服从β分布的样本数据。logL 的长度与数据 data 的长度相同。

[logL,info] = betalike(params,data)还返回 Fisher 信息矩阵 info。info 的对角元素为相应参数的渐进方差。

betalike 是β分布最大似然估计的实用函数。似然假设数据样本中，所有的元素相互独立。因为 betalike 返回负β对数似然函数，用 fmins 函数最小化 betalike 与最大似然估计的功用是相同的。

举例：本例所取数据是随机产生的β分布数据。
```
r = betarnd(4,3,100,1);
[logl,info] = betalike([3.9010  2.6193], r)
logl =
     -33.0514
```

```
info =
      0.2856 0.1528
      0.1528 0.1142
```

参见：betafit, fmins, gamlike, mle, weiblike。

2 gamlike

功能：γ分布的负对数似然函数。

格式：logL = gamlike(params,data)

[logL,info] = gamlike(params,data)

说明：logL = gamlike(params, data)返回由给定样本数据 data 确定的γ分布的参数 params（即[a,b]）的负对数似然函数值。矢量 logL 的长度与 data 矢量的长度相同。

[logL,info] = gamlike(params,data)则同时给出了 Fisher 信息矩阵 info。info 矩阵的对角元素为相应参数的渐进方差。

gamlike 是γ分布的最大似然估计的工具函数。因为 gamlike 返回负γ对数似然函数值，故用 fmins 函数将 gamlike 最小化后，其结果与最大似然估计是相同的。

举例：

```
a = 2; b = 3;
r = gamrnd(a,b,100,1);
[logL,info] = gamlike([2.1990 2.8069],r)
logL =
      267.5585
info =
      0.0690    －0.0790
     －0.0790     0.1220
```

参见：betalike, fmins, gamfit, mle, weiblike。

3 normlike

功能：正态分布的负对数似然函数。

格式：L = normlike (params,data)

说明：与 betalike 和 gamlike 的功能类似，不再赘述。params 参数中，params(1)为正态分布的参数μ， params(2) 为参数σ。

参见： MLE。

4 weiblike

功能：威布尔分布的负对数似然函数。

格式：logL = weiblike(params,data)

[logL,info] = weiblike(params,data)

说明：威布尔分布的负对数似然函数定义为

$$-\log L = -\log\prod_{i=1}^{n} f(a,b\mid x_i) = -\sum_{i=1}^{n}\log f(a,b\mid x_i) \qquad (1.3.2)$$

logL = weiblike(params,data)返回参数为 a、b 的威布尔分布 (params(1)=a, params(2)=b)在给定数据 x_i (即 data)时的负对数似然值。

[logL,info] = weiblike(params,data)则增加了 Fisher 信息量矩阵 info。info 的对角元素为相应参数的渐进方差。

举例：
```
r = weibrnd(0.5,0.8,100,1);
[logL,info] = weiblike([0.4746 0.7832],r)
logL =
      203.8216
info =
      0.0021 0.0022
      0.0022 0.0056
```

参见：betalike, gamlike, mle, weibfit。

表 1.3.2　参数估计函数

分布名称	函数名称	函数调用格式	说　明
最大似然估计	mle	phat = mle('dist',data) [phat,pci] = mle('dist',data) [phat,pci] = mle('dist',data,alpha) [phat,pci] = mle('dist',data,alpha,p1)	'dist'：函数名；　data：数据样本；　alpha：置信度 phat：参数估计结果；　pci：置信区间计算结果 表中其他各函数中的 x、alpha 、phat 和 pci 的含义与此相同
β 分布	betafit	phat = betafit(x) [phat,pci] = betafit(x,alpha)	x：数据样本；phat：参数 a、b 的估计结果矢量[a,b]。置信区间结果 pci 以 2 × 2 矩阵的形式给出，其第一列为参数 a 的置信下界和上界，第二列为参数 b 的置信下界和上界，以下类同
二项分布	binofit	phat = binofit(x,n) [phat,pci] = binofit(x,n) [phat,pci] = binofit(x,n,alpha)	n ：总试验次数 phat：试验成功的概率参数 p 的估计值
指数分布	expfit	muhat = expfit(x) [muhat,muci]= expfit(x) [muhat,muci] = expfit(x,alpha)	muhat：参数 μ 的估计值 muci：参数 μ 估计的置信区间
γ 分布	gamfit	phat = gamfit(x) [phat,pci] = gamfit(x) [phat,pci] = gamfit(x,alpha)	phat：参数 a、b 的估计值[a ,b] pci：参数 a、b 估计的置信区间
正态分布	normfit	[muhat,sigmahat,muci,sigmaci]= normfit(X) [muhat,sigmahat,muci,sigmaci]=normfit(X,alpha)	muhat：参数 μ 的估计值；muci：参数 μ 估计的置信区间 sigmahat：参数 σ 的估计值；sigmaci：参数 σ 估计的置信区间
泊松分布	poissfit	lambdahat = poissfit(X) [lambdahat,lambdaci] = poissfit(X) [lambdahat,lambdaci] = poissfit(X,alpha)	lambdahat：参数 λ 的估计值 lambdaci：参数 λ 估计的置信区间
均匀分布	unifit	[ahat,bhat] = unifit(X) [ahat,bhat,ACI,BCI] = unifit(X) [ahat,bhat,ACI,BCI] = unifit(X,alpha)	ahat：参数 a 的估计值；　ACI：参数 a 估计的置信区间 bhat：参数 b 的估计值；BCI：参数 b 估计的置信区间
威布尔分布	weibfit	phat = weibfit(x) [phat,pci] = weibfit(x) [phat,pci] = weibfit(x,alpha)	phat：参数 a、b 的估计值[a ,b] pci：参数 a、b 估计的置信区间

1.4 描述性统计

1.4.1 概述

数据样本少则几个，多则成千上万，人们希望能用少数几个包含其最多相关信息的数值来体现数据样本总体的规律。描述性统计就是搜集、整理、加工和分析统计数据，使之系统化、条理化，以显示出数据资料的趋势、特征和数量关系。它是统计推断的基础，实用性较强，在统计工作中经常使用。

根据统计量特征性质的不同，工具箱提供了位置度量、散布度量、自助法以及在缺失数据情况下处理方法等方面的描述统计工具函数，见表 1.4.1。

表 1.4.1 描述性统计函数表

函数分类	函数名称	功能
位置度量	geomean	几何均值
	harmmean	调和均值
	mean	算术平均值
	median	中位数
	trimean	修正的样本均值
散布度量	var	方差
	iqr	内四分位数间距
	mad	平均绝对偏差
	range	样本极差
	std	标准差
	moment	任意阶中心矩
	cov	协方差矩阵
缺失数据情况下的统计处理函数	nanmax	忽视缺失数据的最大值
	nanmean	忽视缺失数据的平均值
	nanmedian	忽视缺失数据的中位数
	nanmin	忽视缺失数据的最小值
	nanstd	忽视缺失数据的标准差
	nansum	忽视缺失数据的和
相关系数	corrcoef	计算相关系数
样本分位数	prctile	样本的经验分位数
样本峰度	kurtosis	计算样本峰度
样本偏度	skewness	计算样本偏度
自助法	bootstrp	通过对数据重新采样进行自助统计

1.4.2 中心趋势(位置)度量

数据样本中心趋势度量的目的在于对数据样本在数据分布线上分布的中心予以定位，即中心位置的度量。

均值是对位置简单和通常的估计量。如果数据样本来自正态分布，样本均值是最优估计量(μ的最小方差无偏估计)。不幸的是，几乎所有的实际数据中都存在野值，这通常是数据输入错误或其他小的技术问题所造成的。样本均值对这些问题是敏感的。一个糟糕的数据的加入可能使样本均值偏离正常值很远。

中位数和修正的均值则受野值的干扰很小。中位数是样本的 50%分位点，如果加入一个任意大的干扰值，中位数仅可能有很小的移动。而修正的均值所蕴含的思想是剔除样本中部分最高值和最低值来决定样本的中心位置。

几何均值和调和均值对野值都较敏感。当样本服从对数正态分布或偏斜程度很大时，它们也都是有效的方法。

下例显示上述几种统计量对野值的敏感程度。

```
x = [ones(1,6) 100]
x =
    1  1  1  1  1  1  100
locate = [geomean(x) harmmean(x) mean(x) median(x)?
        trimmean(x,25)]
locate =
    1.9307  1.1647  15.1429  1.0000  1.0000
```

位置度量有关函数的详细说明如下。

1 geomean

功能：样本的几何均值。

格式：m = geomean(X)

说明：几何均值的定义为

$$m = \left[\prod_{i=1}^{n} x_i \right]^{\frac{1}{n}} \tag{1.4.1}$$

geomean 函数计算样本的几何均值。X 若为矢量，geomean(X)返回 X 中元素的几何均值；X 若为矩阵，geomean(X)给出的结果为一个行矢量，其每个元素为 X 的对应列元素的几何均值。

举例：样本均值大于或等于其几何均值。

```
x = exprnd(1,10,6);
geometric = geomean(x)
geometric =
     0.7466 0.6061 0.6038 0.2569 0.7539 0.3478
average = mean(x)
average =
     1.3509 1.1583 0.9741 0.5319 1.0088 0.8122
```

参见：mean, median, harmmean, trimmean。

2 harmmean

功能：样本数据的调和均值。

格式：m = harmmean(X)

说明：样本的调和均值定义为

$$m = \frac{n}{\sum_{i=1}^{n} \frac{1}{x_i}} \qquad (1.4.2)$$

harmmean(X)计算数据样本 X 的调和均值。X 若为矢量，harmmean(X)返回 X 中元素的调和均值；X 若为矩阵，harmmean(X)给出的结果为一个行矢量，其每个元素为 X 的对应列元素的调和均值。

举例：样本均值大于或等于其调和均值。

```
x = exprnd(1,10,6);
harmonic = harmmean(x)
harmonic =
    0.3382 0.3200 0.3710 0.0540 0.4936 0.0907
average = mean(x)
average =
 1.3509 1.1583 0.9741 0.5319 1.0088 0.8122
```

参见：mean, median, geomean, trimmean。

3 mean

功能: 样本数据的算术平均值。

格式: m=mean(X)

说明: mean(X) 计算数据样本X 的算术平均值

$$\overline{x_j} = \frac{1}{n}\sum_{i=1}^{n} x_{ij} \qquad (1.4.3)$$

X 若为矢量，mean(X)返回 X 中元素的算术平均值；X 若为矩阵，mean(X)给出的结果为一个行矢量，其每个元素为 X 的对应列元素的算术平均值。

举例: 以下命令产生 5 组正态分布随机数，每组 100 个数，样本均值为 0，标准差为 1，则 xbar 中的样本均值的变化范围为（0.00±0.10）：

```
x = normrnd(0,1,100,5);
xbar = mean(x)
xbar =
    0.0727 0.0264 0.0351 0.0424 0.0752
```

参见：median, std, cov, corrcoef, var。

4 median

功能：样本数据的中值（即中位数）。

格式：m = median(X)

说明：median(X)计算中值，即数据样本的 50%中位数。中位数是数据样本中心位置的鲁棒估计，因为野值的出现对其影响很小。

X 若为矢量，median(X)返回 X 中元素的中值；X 若为矩阵，median(X)给出的结果为一个行矢量，其每个元素为 X 的对应列元素的中值。因为此函数使用分类检索方法而实现，对于大矩阵，消耗时间和空间多一些。

举例：
```
xodd = 1:5;
modd = median(xodd)
modd =
       3
meven = median(xeven)
meven =
       2.5000
```
下面的计算表明了野值对计算结果的影响。
```
xoutlier = [x 10000];
moutlier = median(xoutlier)
moutlier =
          3
```
参见：mean, std, cov, corrcoef。

5 trimmean

功能：剔除极端数据的样本均值。

格式：m = trimmean(X,percent)

说明：trimmean(X,percent)计算剔除数据观测量中最高 percent %和最低 percent %的数据后的均值。修正样本均值是样本位置量的鲁棒估计。如果数据中有野值，则修正样本均值是样本数据体的中心更为合适的估计。如果数据全部来自同一概率分布，那么，作为数据样本位置的估计量，修正样本均值不如样本均值有效。

举例：本例给出了 10 %修正样本均值与样本均值之间相对效率的蒙特卡罗仿真结果。
```
x = normrnd(0,1,100,100);
m = mean(x);
trim = trimmean(x,10);
sm = std(m);
strim = std(trim);
efficiency = (sm/strim).^2
efficiency =
             0.9702
```
参见：mean, median, geomean, harmmean。

1.4.3 散布度量

散布度量可以理解为样本中的数据偏离其数值中心的程度，也称离差。

极差(range)，亦称全距，定义为样本最大观测值与最小观测值之差，是对散布最简单的度量，但对野值非常敏感。

标准差(std)和方差(var)为通常的散布度量，它们对于正态分布的样本描述是最优的。样本方差是正态分布参数σ^2的最小方差无偏估计,标准差为方差的平方根。但它们对野值的抗干扰能力都较弱，偏离数据整体的一个数值对统计量的影响可能非常大。

平均绝对偏差(mad)对野值也很敏感，但其程度比方差和标准差要小一些。

四分位数间距(iqr)为随机变量的上四分位数$X_{0.75}$和下四分位数$X_{0.25}$之差：$X_{0.75}-X_{0.25}$是随机变量以1/2的概率取值区间的长度。因为只有中间50%的数据影响其度量值，故其对野值的抗干扰能力很强。

下例显示了这几种统计量对野值的敏感程度。

```
x = [ones(1,6) 100]
x =
       1  1  1  1  1  1  100
stats = [iqr(x) mad(x) range(x) std(x)]
stats =
       0   24.2449   99.0000   37.4185
```

散布度量函数的详细说明如下。

1 iqr

功能：计算样本的内四分位数间距。

格式：y = iqr(X)

说明：iqr(X)计算数据样本X的75%和25%分位数之差。iqr是数据的散度的鲁棒估计，因为25%~75%分位数外的数据变化对iqr没有影响。如果数据有野值，则作为数据整体散度的估计，iqr比标准差更具代表性。而当数据全部来自正态分布时，作为散度的估计，iqr不如标准差有效。正态分布的标准差σ可用iqr乘以0.7413作为估计量。

举例：本例采用蒙特卡罗仿真方法来说明正态分布的iqr与标准差之间的相对效率。

```
x = normrnd(0,1,100,100);
s = std(x);
s_IQR = 0.7413 *iqr(x);
efficiency = (norm(s-1)./norm(s_IQR-1)).^2
efficiency =
0.3297
```

参见：std, mad, range。

2 mad

功能：样本数据的平均绝对偏差。

格式：y = mad(X)

说明：mad(X)计算一组数据及其样本均值间的绝对偏差的平均值。X 若为矢量，mad(X)返回 X 中元素的平均绝对偏差；X 若为矩阵，mad(X)给出的结果为一个行矢量，其每个元素为 X 的对应列元素的平均绝对偏差。当数据全部来自正态分布时，作为散度的估计，mad 不如标准差有效。正态分布的标准差σ，可用 mad 乘以 1.3 作为估计量。

举例：本例采用蒙特卡罗仿真方法来说明正态分布的 mad 与标准差之间的相对效率。

```
x = normrnd(0,1,100,100);
s = std(x);
s_MAD = 1.3 *mad(x);
efficiency = (norm(s-1)./norm(s_MAD-1)).^2
efficiency =
0.5972
```

参见：std, range。

3 range

功能：计算样本极差。

格式：y = range(X)

说明：range(X)返回样本数据的最大值和最小值之差。X 若为矢量，range(X)返回 X 中数据元素的极差；X 若为矩阵，range(X)给出的结果为一个行矢量，其每个元素为 X 的对应列的极差。极差是样本散度的简单估计算法，当出现野值时，往往估计不准。

举例：标准正态分布的大样本数据的极差大约为 6。统计质量管理应用中的工序性能指数 C_p 和 C_{pk} 即由此而来。

```
rv = normrnd(0,1,1000,5);
near6 = range(rv)
near6 =
    6.1451 6.4986 6.2909 5.8894 7.0002
```

参见：std, iqr, mad。

4 var

功能：计算样本方差。

格式：y = var(X)

　　　y = var(X,1)

y = var(X,w)

说明：var(X)计算 X 中数据的方差。X 若为矢量，var(X)返回 X 中数据元素的方差；X 若为矩阵，var(X)给出的结果为一个行矢量，其每个元素为 X 的对应列的方差。var(X)是经 $n-1$ 进行了标准化，其中，n 是数据长度。对于正态分布，这使 var(X)成为 σ^2 的最小方差无偏估计量(MVUE)。

y = var(X,1) 是经 n 进行了标准化,得到关于其均值(惯性矩)的样本数据的二阶矩。

y = var(X,w)使用权重矢量 w 计算方差。w 中元素的个数必须与 X 的行数相同。若 X 是矢量，X 和 w 的长度应该匹配。w 中每个元素必须为正数。var 支持方差两种通常的定义：设 ss 为 X 中数据与均值之差的平方和，则 $\mathrm{var(x)} = \mathrm{SS}/(\mathrm{n}-1)$ 为方差的 MVUE，$\mathrm{var(x,1)} = \mathrm{SS}/\mathrm{n}$ 为方差的最大似然估计（MLE）。

举例：
```
x = [-1 1];
w = [1 3];
v1 = var(x)
v1 =
2
v2 = var(x,1)
v2 =
1
v3 = var(x,w)
v3 =
0.7500
```

参见：cov, std。

5 std

功能：样本的标准差。

格式：y = std(X)

说明：标准差的定义为

$$s = \left(\frac{1}{n-1}\sum_{i=1}^{n}(x_i - \bar{x})^2\right)^{\frac{1}{2}} \quad \bar{x} = \frac{1}{n}\sum x_i \tag{1.4.4}$$

由上式可知，std(X)是经 $n-1$ 进行了标准化，其中，n 是数据长度。对于正态分布，标准差的平方是 σ^2 的最小方差无偏估计量 std(X)返回数据样本的标准差。X 若为矢量，std(X)返回 X 中数据元素的标准差；X 若为矩阵，std(X)给出的结果为一个行矢量，其每个元素为 X 的对应列的标准差。

举例：例中，X 的每列的 y 的期望值为 1。

```
x = normrnd(0,1,100,6);
y = std(x)
y =
```

```
    0.9536 1.0628 1.0860 0.9927 0.9605 1.0254
y = std(-1:2:1)
y =
    1.4142
```

参见：cov, var。

6　cov

功能：协方差矩阵。

格式：C = cov(X)

C = cov(x,y)

说明：cov 函数是 Matlab 工具箱中所定义的函数，用于计算协方差。X 若为单个矢量，cov(X)返回包含方差的标量；X 若为矩阵，X 的每一列表示一个变量而行元素为观测值。cov(X)计算结果为协方差矩阵。方差函数 var(X)与 diag(cov(X))所返回的结果相同；标准差函数 std(X)等价于 sqrt(diag(cov(X)))。cov(x,y) (其中 x 和 y 为等长度的列矢量)与 cov([x　y])的计算结果相同。

算法：cov 函数的算法为：

```
[n,p] = size(X);
X = X -ones(n,1) *mean(X);
Y = X' *X/(n-1);
```

参见：corrcoef, mean, std, var 和信号处理工具箱中的 xcov, xcorr。

1.4.4　处理缺失数据的函数

在对大量的数据样本进行处理分析时，常会遇到一些数据无法找到或不能确定某个数据的确切值的情况。在这种情况下，我们以符号“NaN”（not a number）标注这样的数据。工具箱中的一些描述统计函数（见表 1.4.2）在进行统计计算时可自动处理包含 NaN 数据的样本。

例如，m 中包含缺失（NaN）数据。

```
m = magic(3);
m([1 5 9]) = [NaN NaN NaN]
m =
    NaN      1       6
     3      NaN      7
     4       9      NaN
```

用一般的函数可以说得不到任何信息，如用 sum 函数

```
sum(m)
ans =
      NaN   NaN   NaN
```

通过能够处理缺失数据的函数则可进行特殊处理，得到有用的信息。

```
nansum(m)
ans =
      7   10   13
```

调用处理缺失数据的函数的方法基本上是相同的。下面以 nanmax 为例，详细说明其调用方法。其他函数的功能和调用格式及说明参见表 1.4.2。

- nanmax

功能：忽视 NaN，求其他数据的最大值。

格式：m = nanmax(X)

[m,ndx] = nanmax(X)

m = nanmax(a,b)

说明：m = nanmax(X)计算存在 NaN 标记的数据样本 X 中的最大值。X 若为矢量，nanmax(X)返回 X 中除 NaN 外的其他元素中的最大值；X 若为矩阵，nanmax(X)给出的结果为一个行矢量，其每个元素为 X 的对应列非 NaN 元素的最大值。[m,ndx] = nanmax(X)还将数据的大小序号返回矢量 ndx 中。m = nanmax(a,b)返回 a 或 b 的最大数。注意，a 和 b 的长度应该相称。

举例：
```
m = magic(3);
m([1 6 8]) = [NaN NaN NaN]
m =
  NaN  1    6
  3    5    NaN
  4    NaN  2
[nmax,maxidx] = nanmax(m)
nmax =
  4 5 6
maxidx =
  3 2 1
```

参见：nanmin, nanmean, nanmedian, nanstd, nansum。

表 1.4.2　描述统计中处理缺失数据的函数

函 数 名 称	功　能	调 用 格 式
nanmax	求包含缺失数据的样本的最大值	m = nanmax(X) [m,ndx] = nanmax(X) m = nanmax(a,b)
nanmin	求包含缺失数据的样本的最小值	m = nanmin(X) [m,ndx] = nanmin(X) m = nanmin(a,b)
nanmean	求包含缺失数据的样本的平均值	y = nanmean(X)
nanmedian	求包含缺失数据的样本的中位数	y = nanmedian(X)
nanstd	求包含缺失数据的样本的标准差	y = nanstd(X)
nansum	求包含缺失数据的样本的和	y = nansum(X)

1.4.5　中心矩

中心矩是关于数学期望的矩。对于任意 $r\geqslant 0$,称 $\mu=E(X-EX)^r$ 为随机变量 X 的 r 阶中心矩。一阶中心矩为 0，二阶中心矩为方差：$\mu_2=DX$。

- moment

  功能：任意阶中心矩。

  格式：m = moment(X,order)

  说明：一种分布的 k 阶中心矩定义为

$$m_n = E(x-\mu)^k \tag{1.4.5}$$

  $E(x)$为 x 的期望值。

  Moment(X,order)返回由正整数 order 所确定阶的 X 的中心矩。X 若为矢量，moment(X,order)返回 X 中元素的中心矩；X 若为矩阵，moment(X,order)给出的结果为一个行矢量，其每个元素为 X 的对应列元素的中心矩。注意，一阶中心矩为 0，二阶中心矩为方差。n 为数据矢量 X 的长度，函数本身计算时除数为 n，而非 n－1。

  举例：
```
X = randn([6 5])
X =
  1.1650    0.0591    1.2460   -1.2704    -0.0562
  0.6268    1.7971   -0.6390    0.9846     0.5135
  0.0751    0.2641    0.5774   -0.0449     0.3967
  0.3516    0.8717   -0.3600   -0.7989     0.7562
 -0.6965   -1.4462   -0.1356   -0.7652     0.4005
  1.6961   -0.7012   -1.3493    0.8617    -1.3414
 m = moment(X,3)
 m =
 -0.0282    0.0571    0.1253    0.1460    -0.4486
```

  参见：kurtosis, mean, skewness, std, var。

1.4.6　百分位数及其图形描述

尝试用位置度量和散布度量两种参量来描述数据样本是简捷的，但或许会产生误导。另一种选择是计算样本百分位数的一个合理的数值，它不仅提供数据形态的信息，也给出它的位置和散布信息。

举例 ：总的样本包含来自两个不同正态分布的数据。

```
x = [normrnd(4,1,1,100) normrnd(6,0.5,1,200)];
```

```
p = 100*(0:0.25:1);
y = prctile(x,p);
z = [p; y']
z =
     0    25.0000    50.0000    75.0000    100.0000
  1.5172   4.6842     5.6706    6.1804     7.6035
```

用 Box 图(图 1.4.1)来表现上面的结果。

```
boxplot(x)
```

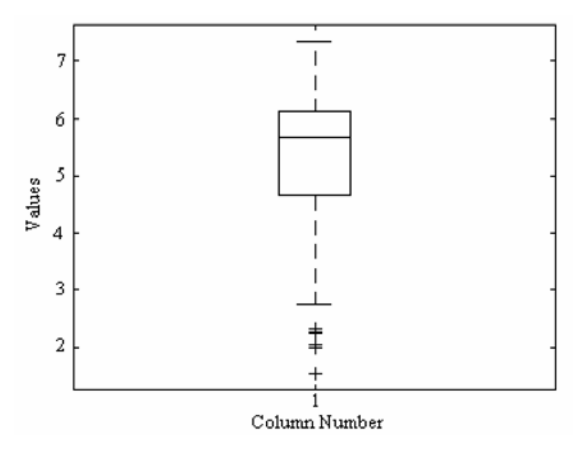


图 1.4.1 两组不同正态分布数据的 Box 图形

下面的“长尾巴”和符号“＋”表明样本数值缺乏对称性。关于 Box 图的详细说明可参见第 1.6 节。另外用直方图(图 1.4.2)给出上述样本，以进一步认识。

```
Hist(x)
```

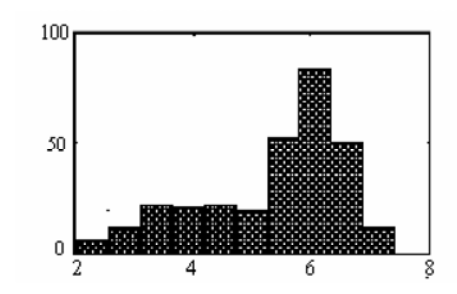


图 1. 4. 2 两组不同正态分布数据的直方图

关于样本的百分位数计算函数 prctile 的详细说明如下。

- prctile

  功能：计算样本的百分位数。

  格式：Y = prctile(X,p)

  说明：Y = prctile(X,p)计算数据 X 中大于 p%的值，p 的取值区间为[0,100]。X 若为矢量，prctile(X,p)返回 X 中 p 百分位数；X 若为矩阵，prctile(X,p)给出的结

果为一个行矢量，其每个元素为 X 的对应列 p 百分位数。如果 p 是矢量，则 Y 的第 i 行对应于 X 的 p(i)百分位数。

举例：
```
x = (1:5)'*(1:5)
x =
 1 2 3 4 5
 2 4 6 8 10
 3 6 9 12 15
 4 8 12 16 20
 5 10 15 20 25
y = prctile(x,[25 50 75])
y =
 1.7500 3.5000 5.2500  7.0000   8.7500
 3.0000 6.0000 9.0000 12.0000 15.0000
 4.2500 8.5000 12.7500 17.0000 21.2500
```

1.4.7　相关系数

相关系数亦称“单相关系数”、“全相关系数”、“乘积矩相关系数”等，是两个随机变量间线性相依程度的度量。

- corrcoef

功能：相关系数。

格式：R = corrcoef(X)

说明：corrcoef 是 Matlab 工具箱中所定义的函数。

输入矩阵 X 的行元素为观测值，列元素为变量，R = corrcoef(X)返回相关系数矩阵 R。矩阵 R 的元素 R（i,j）与协方差矩阵 C（＝cov(x)）的对应元素之间的关系可由下式表示

$$rR(i,j)=\frac{C(i,j)}{\sqrt{C(i,i)C(j,j)}} \tag{1.4.6}$$

参见：cov, mean, std, var。

1.4.8　样本峰度和样本偏度

(1)　样本峰度

峰度为单峰分布曲线“峰的平坦程度”的度量。峰度的定义为

$$k = \frac{E(x-\mu)^4}{\sigma^4} \tag{1.4.7}$$

其中，$E(x)$为 x 的期望值。注意：关于峰度的定义，不同文献有所不同，一般定义为(1.4.7)式计算值减 3，此时正态分布具有 0 峰度。在本工具箱中的峰度函数不使用此惯例。正态分布的峰度为 3。曲线比正态分布曲线平坦的分布，其峰度大于 3；反之，则小于 3。

- kurtosis

功能：样本峰度。

格式：k = kurtosis(X)

说明：k = kurtosis(X)返回数据样本 X 的峰度。X 若为矢量，kurtosis(X)返回 X 中元素的峰度；X 若为矩阵，kurtosis(X)给出的结果为一个行矢量，其每个元素为 X 的对应列元素的峰度。

举例：
```
X = randn([5 4])
X =
  1.1650   1.6961  -1.4462  ? .3600
  0.6268   0.0591  -0.7012  -0.1356
  0.0751   1.7971  1.2460   -1.3493
  0.3516   0.2641  -0.6390  -1.2704
  －0.6965 0.8717   0.5774   0.9846
k = kurtosis(X)
k =
  2.1658  1.2967  1.6378  1.9589
```

参见：mean, moment, skewness, std, var。

(2) 样本偏度

偏度是样本数据围绕其均值的对称情况的度量。如果偏度为负，则数据分布偏向于其均值的左边；反之，则偏向右边。样本分布的偏度定义为

$$y = \frac{E(x-\mu)^3}{\sigma^3} \tag{1.4.8}$$

$E(x)$是 x 的期望。

正态分布（或其他相当对称的分布）的偏度为 0。

- skewness

功能：样本偏度。

格式：y = skewness(X)

说明：skewness(X)返回数据样本的偏度。X 若为矢量，skewness(X)返回 X 中数据元素的偏度；X 若为矩阵，skewness(X)给出的结果为一个行矢量，其每个元

素为 X 的对应列的偏度。

举例：
```
X = randn([5 4])
X =
   1.1650 1.6961 -1.4462 -0.3600
   0.6268 0.0591 -0.7012 -0.1356
   0.0751 1.7971 1.2460 -1.3493
   0.3516 0.2641 -0.6390 -1.2704
   -0.6965 0.8717 0.5774 0.9846
y = skewness(X)
y =
   -0.2933 0.0482 0.2735 0.4641
```

参见：kurtosis, mean, moment, std, var。

1.4.9　自助法(Bootstrap)

在近 10 年中，不少统计文献研究了通过重新采样数据，获取有关统计估计量不确定性信息的性质。自助法实施的过程涉及从一个数据库中选取和用同样的方法分析每一个复位样本。所谓复位样本就是每一个从数据库中采集的样本在采样完毕后都归还到数据库中。每次用于 bootstrap 采样的样本元素即为数据库中的初始元素。因此，某一来自初始数据库的特定数据能够在一个给定的 bootstrap 样本中出现多次。所获得的样本估计量的范围能使我们掌握所估计的量的不确定度。

图 1.4.3 的数据来自对 15 个法律学校学生的 LAST 分数和 GPA 进行比较的样本。

```
load lawdata
plot(lsat,gpa,'+')
lsline
```

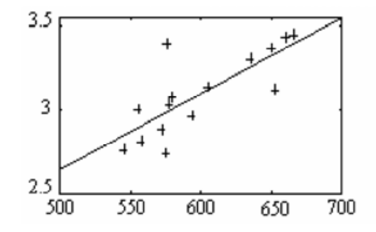


图 1.4.3　LAST 分数和 GPA 的最小二乘拟合线

图 1.4.3 中的最小二乘拟合线表明，LAST 分数随着 GPA 的提高而增长。但我们确信此结论的程度是多少？曲线只给出了直观的表现，但没有量的表示。我们可以用 corrcoef 函数来计算两变量间的相关系数。

```
rhohat=corrcoef(lsat,gpa)
rhohat =
   1.0000    0.7764
```

```
    0.7764    1.0000
```

结果表明，LAST 分数与 GPA 变量间存在确定的联系，二者的相关系数为 0.7764。0.7764 看起来较大，但我们仍然不能把握在统计上其显著性有多大。因此采用 bootstrp 函数对 LAST 分数和 GPA 的样本重新采样多次，并考察相关系数的变化情况。

```
rhos1000=bootstrp(1000,'corrcoef',lsat,gpa);
```

上面的命令对 LAST 分数和 GPA 数据实施 1000 次采样，并计算每组样本中二者的相关系数，计算结果的直方图表示如图 1.4.4 所示。

```
hist(rhos1000(:,2),30)
```

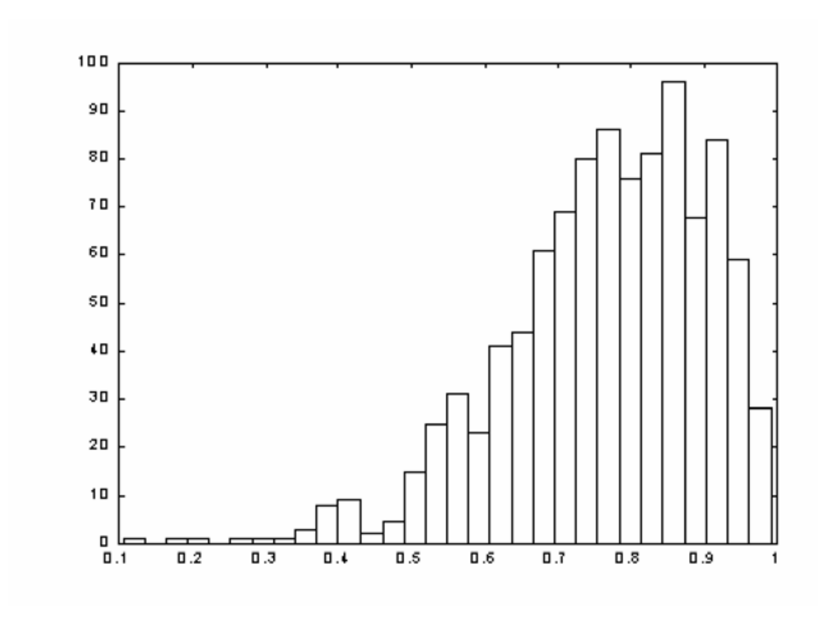


图 1.4.4　LAST 分数和 GPA 数据相关系数直方图

结果显示，相关系数的绝大多数在区间[0.4,1.0]内，表明 LAST 分数和 GPA 具有确定的相关性。而且，进行这样的分析不需要对相关系数的概率分布做出很强的假设。

- bootstrp

功能：通过对数据重新采样进行自助统计。

格式：bootstat = bootstrp(nboot,'bootfun',d1,...)

[bootstat,bootsam] = bootstrp(...)

说明：bootstrp(nboot,'bootfun',d1,...)对数据样本采样 nboot 次，每次采用‘bootfun’函数(bootfun 为用户确定的函数)进行分析。nboot 必须是正整数。d1，d2…为‘bootfun’函数所需的参数。

[bootstat,bootsam] = bootstrp(...)将统计结果返回 bootstat。bootstat 的每一行是 bootstat 每次采样并进行分析后的结果。如果每次分析的结果为一矩阵，则输出自动转换为一矢量并保存在 bootstat 中。bootsam 给出被采样数据在数据库中的序数矩阵。

1.5 假设检验

1.5.1 概述

假设检验是统计推断的基本问题之一，主要是确定关于样本总体特征的判断是否合理的过程。按一定规则(检验准则)根据样本所作假设 H 是否成立，以决定是接受还是否定 H。假设检验的判断和结论是根据样本做出的，故具有“概率性”。

例如，马萨诸塞州无铅汽油的平均价格为每加仑 1.15 美元。怎样决定这个说法是正确的呢？您当然可以调查，州内每个加油站出售了多少油，价格分别是多少？这种方法当然是确切的，但用此方法来获得信息并不一定值得。

一个简单的方法就是在州内随机抽取少数的加油站，查明其价格以及销售量，将统计结果与 1.15 美元的价格相比较。当然，所统计的平均价格或许恰好等于 1.15，但绝大多数情况下，是有差异的。如果算出来的平均价格为 1.18,那么，这是否就说明原来的假设就不正确呢？假设检验就是解决此类问题的方法。

首先对几个必要的名词作简要的解释。

- 零假设——即初始判断。此例中，零假设是平均价格每加仑 1.15 美元，表达为

$$H_0:\ \mu=1.15 \tag{1.5.1}$$

对此有三种可能的备选假设（也称对立假设）

$$H_1:\ \mu>1.15\quad;\quad H_1:\ \mu<1.15\quad;\quad H_1:\ \mu\neq 1.15 \tag{1.5.2}$$

- 显著性水平——是与支持对立假设而拒绝零假设的确定度有关的量。在小样本的前提下，不能肯定自己的结论，所以事先约定，如果观测到符合零假设的样本值的概率小于显著性水平α，则拒绝零假设。典型的显著性水平$\alpha=0.05$。如果要减少犯错误的可能，可取更小的值。
- p-值——在假定零假设为真的条件下，观测给定样本结果的概率值。如果 p-值小于α，则拒绝零假设。反之则并非如此，如果 p-值大于α，则并不能肯定就接受零假设。此时，您只是没有足够的证据来拒绝零假设。

假设检验的输出包括置信区间。不严格地说，置信区间是那些包含真实的假设量的可选概率的值的范围。例如，假设 1.15 在均值为 μ 的 95%置信区间内，即意味着在 0.05 的显著性水平下，不能拒绝零假设。相反，如果 100（1－α）%置信区间内并不包含 1.15，那么，在α显著性水平下，可以拒绝零假设。

假设检验过程的差别经常来源于研究者对于数据样本所做的假设之不同。Z 检验假设数据为同一正态分布的独立样本，并且已知标准差。而 t 检验的假设除了通过数据估计标准差而非规定一个已知标准差值外，其他的假设与 Z 检验相同。这两种检验方法具有相关的信噪比。

信号为算术平均值与假设均值之差；噪声为指定的或估计的标准差。

$$Z=\frac{\bar{x}-\mu}{\sigma} \quad 或 \quad T=\frac{\bar{x}-\mu}{s} \tag{1.5.3}$$

其中，$\bar{x}=\sum_{i=1}^{n}\frac{x_i}{n}$

如果零假设为真，则 Z 服从标准正态分布 N(0)。T 服从自由度为 ν 的学生 t 分布，其中，ν 等于数据个数减 1。

举例：

使用汽油价格数据 gas.mat。有两组样本，每组 20 个数据，分别为 1993 年一月和二月的价格。

```
load gas
prices = [price1 price2]
prices =
  119 118;  117 115;  115 115;  116 122;  112 118;
  121 121;  115 120;  122 122;  116 120;  118 113;
  109 120;  112 123;  119 121;  112 109;  117 117;
  113 117;  114 120;  109 116;  109 118;  118 125
```

假设马萨诸塞州各加油站汽油的价格标准差为每加仑 4 美分。用 T 检验来判断零假设：一月份每加仑汽油的平均价格为 1.15 美元。

```
[h,pvalue,ci] = ztest(price1/100,1.15,0.04)
  h =
   0
pvalue =
   0.8668
ci =
   1.1340 1.1690
```

检验结果给出布尔值 h。当 h＝0，则不能拒绝零假设。检验结果表明，1.15 美元是合理的。95％置信区间明显包含了＄1.15。

二月份如何？下面用 T 检验来判断。现在假设不知价格的标准差。

```
[h,pvalue,ci] = ttest(price2/100,1.15)
h =
   1
pvalue =
   4.9517e-04
ci =
   1.1675 1.2025
```

结果给出 h＝1。在 0.05 显著性水平下，应拒绝零假设。由结果也可以看出，＄1.15 并非二月份的合理价格，95％置信区间的下限大于 1.15。

函数 ttest2 提供了比较两个样本均值的功能。

```
[h,sig,ci] = ttest2(price1,price2)
h =
    1
sig =
    0.0083
ci =
    －5.7845 －0.9155
```

置信区间的结果表明，一月的价格比二月的价格低 1～6 美分。下面给出了相应的 Box 图(图 1.5.1)。

```
boxplot(prices,1)
set(gca,'XtickLabels',str2mat('January','February'))
xlabel('Month')
ylabel('Prices ($0.01)')。
```

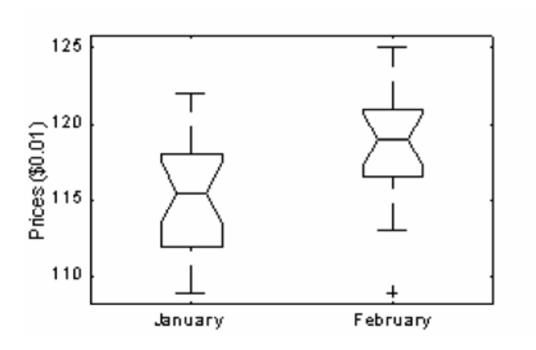


图 1.5.1　汽油价格的 Box 图

1.5.2　函数详解

统计工具箱提供了常用的几种检验方法的函数，见表 1.5.1 。

表 1.5.1　假设检验函数表

函 数 名 称	功　能
ranksum	威尔科克秩和检验
signrank	威尔科克符号秩检验
signtest	成对样本的符号检验
ttest	单样本 t 检验
ttest2	双样本 t 检验
ztest	z 检验

1　ranksum

功能：两个总体一致性的威尔科克秩和检验。

格式：p = ranksum(x,y,alpha)

[p,h] = ranksum(x,y,alpha)

说明：p = ranksum(x,y,alpha)返回两个总体样本 x 和 y 为一致的显著性概率。x 和 y 都是矢量，但长度可以不同；如果二者长度确实不同，则应安排 x 的长度小于 y 的长度。Alpha 为指定的显著性水平，是取值区间为[0，1]的标量。

[p,h] = ranksum(x,y,alpha)返回假设检验的结果 h。如果样本总体 x 和 y 并非明显不一致，则 h＝0；否则，h=1。

举例：对服从泊松分布的两组随机数样本的均值是否相同进行检验。

```
x = poissrnd(5,10,1);
y = poissrnd(2,20,1);
[p,h] = ranksum(x,y,0.05)
p =
   0.0028
h =
   1
```

参见：signrank, signtest, ttest2。

2 signrank

功能：威尔科克符号秩检验。

格式：p = signrank (x,y,alpha)

[p,h] = signrank (x,y,alpha)

说明：p = signrank(x,y,alpha)给出两个配对样本 x 和 y 的中位数相等的假设的显著性概率。矢量 x、y 的长度必须相同，alpha 为给出的显著性水平，取值区间为[0,1]。[p,h] = signrank (x,y,alpha)返回假设检验的结果 h。如果这两个样本的中位数之差几乎为 0，则 h=0；若有显著差异，则 h=1。

举例：检验两个正态分布样本的均值是否相等（首先产生均值相同但标准差不同的随机样本）。注意，对于正态分布，总体的均值和中位数是相同的。

```
x = normrnd(0,1,20,1);
y = normrnd(0,2,20,1);
[p,h] = signrank(x,y,0.05)
p =
  0.2568
h =
  0
```

参见：ranksum, signtest, ttest。

3 signtest

功能：成对样本的符号检验。

格式：p = signtest(x,y,alpha)

[p,h] = signtest(x,y,alpha)

说明：p = signtest(x,y,alpha)也给出假设两个配对样本 x 和 y 的中位数相等的显著性概

率。x 和 y 若为矢量，二者的长度必须相同。y 亦可为标量，在此情况下，计算 x 的中位数与常数 y 之间差异的概率。alpha 为给出的显著性水平, 取值区间为[0,1]。

[p,h] = signtest(x,y,alpha)返回假设检验的结果 h。如果这两个样本的中位数之差几乎为 0，则 h=0；否则若有显著差异，则 h=1。

举例：
```
x = normrnd(0,1,20,1);
y = normrnd(0,2,20,1);
[p,h] = signtest(x,y,0.05)
p =
    0.8238
h =
    0
```

参见：ranksum, signrank, ttest。

4　ttest

功能：单一样本均值的 t 假设检验。

格式：h = ttest(x,m)

h = ttest(x,m,alpha)

[h,sig,ci] = ttest(x,m,alpha,tail)

说明：ttest(x,m)用于正态总体标准差未知时，在 0.05 的显著性水平下，对样本均值是否为 m(即 μ)的 t 检验。

h = ttest(x,m,alpha)与 ttest(x,m)功能是一致的，区别在于自己可选择显著性水平 alpha，而且给出检验结果 h。

[h,sig,ci] = ttest(x,m,alpha,tail)则可在 tail 中规定备选假设，共有三种情形：

tail＝0(缺省) —— $x \neq \mu$

tail＝1 —— $x > \mu$

tail＝-1 —— $x < \mu$

sig 与 T 统计量有关，是在假设 x 的均值等于 μ 时，T 的观测值较大的概率。其中，T 统计量为

$$T=\frac{\bar{x}-\mu}{s}$$

ci 为均值的 1－alpha 置信区间。

举例：给出理论均值为 0、标准差为 1 的 100 个正态随机数样本。当然，观测样本的均值和标准差与理论值是不同的，但假设检验的结果却还原其本质规律。

```
x = normrnd(0,1,1,100);
[h,sig,ci] = ttest(x,0)
h =
   0
sig =
```

```
    0.4474
ci =
   －0.1165 0.2620
```

结果 h=0，意味着我们不能拒绝零假设。

5 ttest2

功能：两个样本均值差异的检验。

格式：[h,significance,ci] = ttest2(x,y)

[h,significance,ci] = ttest2(x,y,alpha)

[h,significance,ci] = ttest2(x,y,alpha,tail)

说明：[h,significance,ci] = ttest2(x,y)检验两个正态分布的样本（x 和 y）是否具有同样的均值，其中假设二者的标准差未知但相等。significance 是与 T 统计量相关的概率值，其中

$$T=\frac{x-y}{s}$$

significance 和 ci 的意义分别与 ttest 中的 sig 和 ci 相同。

[h,significance,ci] = ttest2(x,y,alpha)可自行选择显著性水平 alpha 的大小。

[h,significance,ci] = ttest2(x,y,alpha,tail)中的 tail 给出三种对立假设：

tail＝0 (缺省) —— $\mu_x \neq \mu_y$;

tail＝1 —— $\mu_x > \mu_y$;

tail＝-1 —— $\mu_x < \mu_y$。

举例：首先产生均值为 0，标准差为 1 的 100 个正态分布随机数。然后产生均值为 0.5，标准差为 1 的 100 个正态分布随机数。预定零假设：两个样本的均值相同。

```
x = normrnd(0,1,100,1);
y = normrnd(0.5,1,100,1);
[h,significance,ci] = ttest2(x,y)
h =
    1
significance =
     0.0017
ci =
    -0.7352  -0.1720
```

6 ztest

功能：已知方差的单样本均值的假设检验。

格式：h = ztest(x,m,sigma)

h = ztest(x,m,sigma,alpha)

[h,sig,ci] = ztest(x,m,sigma,alpha,tail)

说明：ztest(x,m,sigma)是在 0.05 显著性水平下检验正态分布的样本（x）是否具有均值 m 和标准差 sigma。

h = ztest(x,m,sigma,alpha)则可由您确定显著性水平 alpha 值，并返回检验结果 h。

[h,sig,ci] = ztest(x,m,sigma,alpha,tail)提供了由 tail 标记的不同对立假设情形的 z 检验：

tail＝0（缺省）—— $x \neq \mu$

tail＝1 —— $x > \mu$

tail＝－1 —— $x < \mu$

sig、ci 与 ttest 函数中相应参数的意义相同。

举例：

```
x = normrnd(0,1,100,1);
m = mean(x)
m =
    0.0727
[h,sig,ci] = ztest(x,0,1)
h =
    0
sig =
    0.4669
ci =
    －0.1232 0.2687
```

1.6 统 计 绘 图

1.6.1 概述

统计工具箱在 Matlab 丰富的绘图功能上又增添了一些特殊的图形表现函数,以充分、直观地展现样本其及统计量的内在规律。box 图用于描述数据样本，也用于通过图形来比较多个样本的均值。正态概率图是确定样本是否为正态分布的图形。分位数-分位数图用于比较两个样本的分布，等等。主要的函数见表 1.6.1。以下各小节详细给出了各函数的使用说明。

表 1.6.1 统计绘图函数表

函 数 名 称	功 能
boxplot	box 图
errorbar	误差条图
fsurfht	函数的交互轮廓图
Gline	交互线绘制
Gname	交互点标记
Lsline	绘制数据图中添加最小二乘拟合线
Normplot	正态概率图
Pareto	帕累托图
Qqplot	分位数-分位数图
Rcoplot	回归残差图
Refcurve	参考多项式
Refline	参考线
Surfht	交互内插轮廓图
Weibplot	威布尔图

1.6.2 box图

如图 1.6.1 所示，box 图为一“盒子”图形，其基本特征为：

- “盒子”的上底和下底间为内四分位间距；“盒子”上下的两条线分别为样本的 25%和 75%分位数。
- “盒子”中间的线为样本中位数。如果中位数不在“盒子”中间，则表明样本存在一定的偏度。
- 虚线贯穿“盒子”上下，显示了样本的其余部分（除非有野值）。假设没有野值，样本的最大值为虚线的顶端；样本的最小值为虚线的底端。缺省时，野值为距离盒子顶端和底端超过 1.5 倍内四分位间距的点，在图中，野值为超出虚线底端的点。
- “＋”表示数据的一个野值。
- “切口”是样本中位数的置信区间。缺省时，box 图是没有切口的。

并排给出的两个带切口的 box 图相当于图形化的 t-检验。

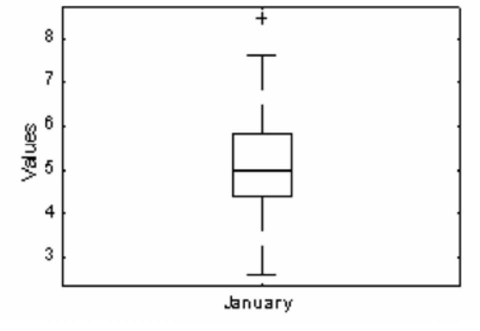


图 1.6.1 box 图

- boxplot

  功能：数据样本的 box 图。

  格式：boxplot(X)

boxplot(X,notch)

boxplot(X,notch,'sym')

boxplot(X,notch,'sym',vert)

boxplot(X,notch,'sym',vert,whis)

说明：boxplot(X)为 X 中的每列数据绘制一个 box 图。

notch 取 1 时，boxplot(X,notch)绘制带切口的 box 图；缺省（即取 notch 为 0）时，box 图无切口。

boxplot(X,notch,'sym')中的‘sym’为野值标记符号(如果有野值的话)。缺省符号为 “＋”。

boxplot(X,notch,'sym',vert)中的 vert 控制 box 图水平放置还是垂直放置。当 vert＝0 时，box 图水平放置，当 vert＝1（缺省）时，box 图垂直放置。

boxplot(X,notch,'sym',vert,whis)中的 whis 定义虚线的长度为内四分位间距(IQR)的函数(缺省情况为 1.5*IQR)。如果 whis＝0，则 box 图用‘sym’规定的标记显示“盒子”外所有的数据。

举例：
```
x1 = normrnd(5,1,100,1);
x2 = normrnd(6,1,100,1);
x = [x1 x2];
boxplot(x,1)
```
图 1.6.2 是二者的 box 图。

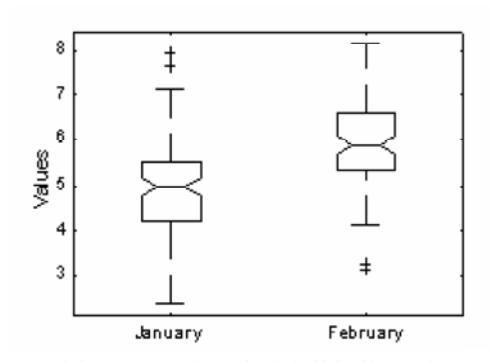


图 1.6.2　两组正态分布数据的 box 图

由图 1.6.2 可见，X 中两列数据的均值之差约为 1。

1.6.3　误差条图

- errorbar

功能：误差条图。

格式：errorbar(X,Y,L,U,symbol)

errorbar(X,Y,L)

errorbar(Y,L)

说明：errorbar(X,Y,L,U,symbol)给出 X～Y 图以及由 L 和 U 规定误差界限的误差条，参见图 1.6.3。误差条是距离点（X,Y）上面的长度为 U（i）、下面的长度为 L(i)的直线。X,Y,L,U 的长度必须相同。如果它们都是矩阵，则每列数据给出一条单独的线。Symbol 为一字符串，可规定线的类型、颜色等。

errorbar(X,Y,L)给出其误差条关于 Y 对称的 X～Y 图。

errorbar(Y,L)绘制区间为[Y-L,Y+L]的误差条图。

举例：

```
lambda = (0.1:0.2:0.5);
r = poissrnd(lambda(ones(50,1),:));
[p,pci] = poissfit(r,0.001);
L = p －pci(1,:)
U = pci(2,:) －p
errorbar(1:3,p,L,U,'+')
L =
    0.1200 0.1600 0.2600
U =
    0.2000 0.2200 0.3400
```

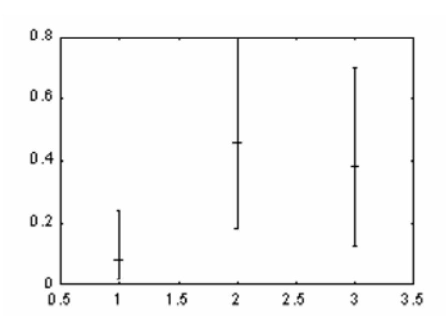


图 1.6.3　泊松分布数据拟合的误差条图

1.6.4　函数的交互轮廓图

- fsurfht

功能：函数的交互轮廓图。

格式：fsurfht('fun',xlims,ylims)

fsurfht('fun',xlims,ylims,p1,p2,p3,p4,p5)

说明：fsurfht('fun',xlims,ylims)给出 fun 函数的交互轮廓图，参见图 1.6.4。其中 fun 为用户指定的函数名称。图中，X 轴的范围由 xlims＝[xmin,xmax]确定，Y 轴的范围由 ylims＝[ymin,ymax]确定。

fsurfht('fun',xlims,ylims,p1,p2,p3,p4,p5)则另外给出 5 个可选参数提供给函数 fun。函数 fun 的前两个参数分别是 X 变量和 Y 变量。

图中分别有一个垂直和水平的参考线（虚线），其交点为当前的 X 值和 Y 值。用户可以拖拉参考线，同时即可观察到更新的 z 值(在图的顶部)。也可在 X

轴和 Y 轴相应的编辑框内输入一定的数而获得所计算的 z 值。

举例：绘制由 gas.mat 文件提供数据的高斯似然函数图形。先写一个 M 文件 gauslike.m。

```
function z = gauslike(mu,sigma,p1)
n = length(p1);
z = ones(size(mu));
for i = 1:n;
z=z.*(normpdf(p1(i),mu,sigma));
end
```

gauslike 函数调用 normpdf 函数，其中视数据样本为固定量而参数 μ 和 ε 为变量。假设汽油价格服从正态分布，现在来看看样本的似然函数曲面。

```
load gas
fsurfht('gauslike',[112 118],[3 5],price1)
```

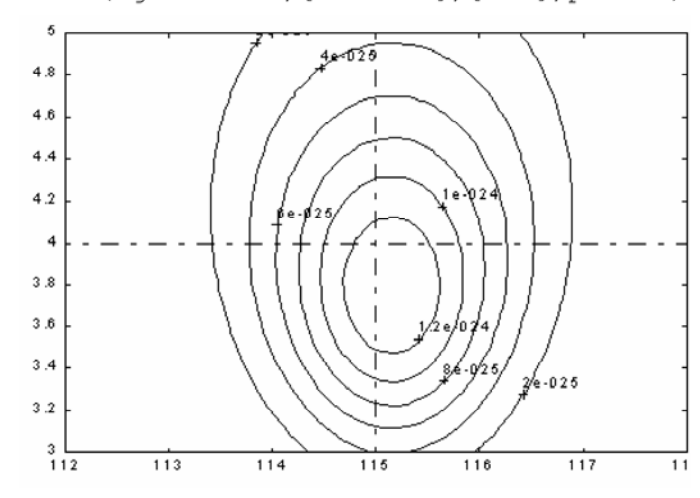


图 1.6.4 高斯似然函数交互轮廓图

其中，样本均值为图中 X 的最大值，但样本标准差并非为图内 Y 的最大值。

```
mumax = mean(price1)
mumax =
   115.1500
sigmamax = std(price1)*sqrt(19/20)
sigmamax =
   3.7719
```

1.6.5 交互线绘制

- gline

  功能：交互线绘制。

格式：gline(fig)

h = gline(fig)

gline

说明：gline(fig)函数使得运用鼠标可在图 fig 中点击两点，绘制出以此两点为端点的直线。点击第一点后，线的一端即固定，另一端为鼠标移动所在的位置，直至点击鼠标时，其所在的位置即为线的另一端所在的位置。

h = gline(fig)返回线段的句柄 h。

gline 则在当前图中绘制线段。

参见：refline, gname。

1.6.6　名称或实例标记

• gname

功能：用实例名称或实例号来标记图中的点。

格式：gname('cases')

gname

h = gname('cases',line_handle)

说明：gname('cases')显示图形窗口(参见图 1.6.5)。图中给出一个十字叉，可用鼠标确定图中需做标记的每一点。具体方法为：用鼠标将十字叉移至某点附近，点击鼠标后，此点就被确定为需做标记的点；同理可用此方法确定多个需做标记的点。然后，当按下 Return 或 Enter 键时，‘cases’中的实例名称就会出现在以上每一点旁。‘cases’是一个字符串矩阵，其每一行是一个数据点的实例名称。不带参数的 gname 函数则用实例号做标记。

h = gname('cases',line_handle)返回由图中文本对象的句柄所组成的矢量。如果图中有不止一条线，数量 line_handle 用于标志相应的线。

举例：使用城市数据库来考察教育和艺术水平最好及最糟的城市。

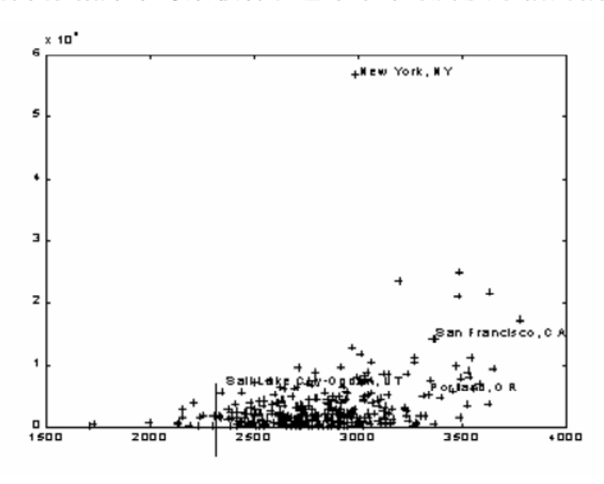


图 1.6.5　名称或实例标记图

```
load cities
education = ratings(:,6); arts = ratings(:,7);
plot(education,arts,'+')
gname(names)
```

参见：gtext。

1.6.7　绘制最小二乘拟合线

- lsline

功能：绘制数据的最小二乘拟合线。

格式：lsline

h = lsline

说明：lsline 为当前坐标系中的每一线性数据组给出其最小二乘拟合线，可参见图 1.6.6。（线条类型为‘—’，‘——’，‘.—’）。

h = lsline 返回线条的句柄。

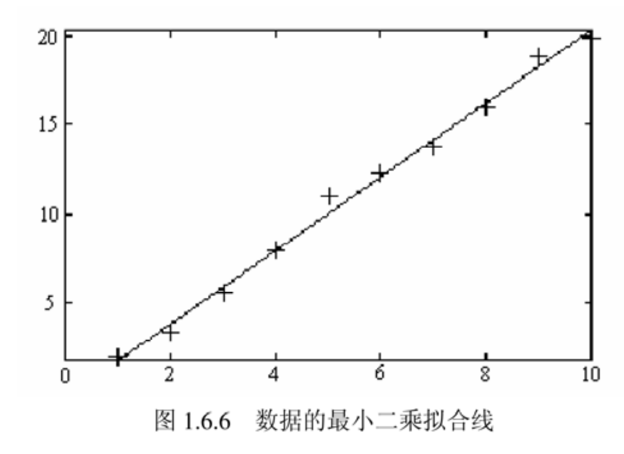


图 1.6.6　数据的最小二乘拟合线

举例：y = [2 3.4 5.6 8 11 12.3 13.8 16 18.8 19.9]';

```
plot(y,'+');
lsline;
```

参见：polyfit, polyval。

1.6.8　正态概率图

- normplot

功能：图形化正态性检验的正态概率图。

格式：normplot(X)

h = normplot(X)

说明：normplot(X)显示数据 X 的正态概率图（参见图 1.6.7）。对于矩阵 X，normplot(X)为其每列显示一条线。图形以符号“＋”标识样本数据。叠加在数据点上的实线是X的每列数据的第一和第三分位间的连线(为样本顺序统计量的鲁棒线性拟合)。此线延伸至样本的两端，有助于评价数据的线性程度。如果数据确实服从正态分布，则图形呈现为直线。而其他概率密度函数则表现出不同的弯曲。

h = normplot(X)返回曲线的句柄。

举例：产生一个正态分布样本并绘制其正态概率图(见图 1.6.7)。

```
x=normrnd(0,1,50,1);
h=normplot(x);
```

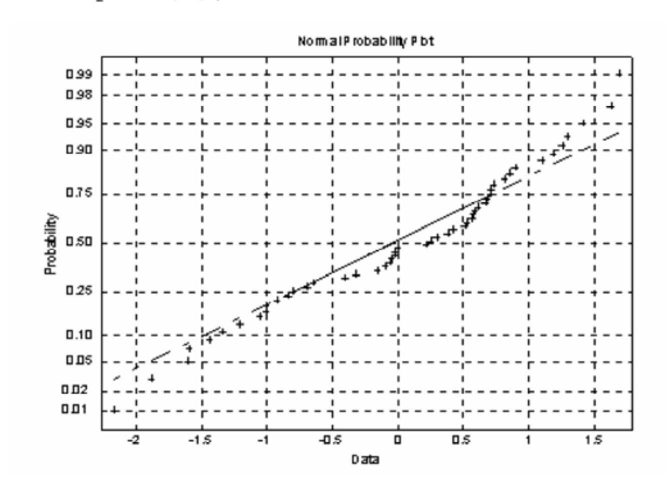


图 1.6.7　一个正态分布样本的正态概率图

1.6.9　帕累托图

- pareto

功能：帕累托图。

格式：pareto(y)

pareto(y,'names')

h = pareto(...)

说明：帕累托图亦称“主次因素排列图”，简称“排列图”，可参见图 1.6.8。 pareto(y)将矢量 y 中的每个元素按数值递减顺序绘成直方条，并以其在 y 中的索引号进行标记。各直方条上方的折线显示累积频率。pareto(y,'names')则以字符串矩阵‘names’中的名称对 y 中相应的元素所绘的直方条进行标记。h = pareto(...)还返回图形的句柄。

举例：某工厂生产了一批零件，并统计了其中存在缺陷的各种零件相应的数量，将其帕累托图绘制出来(见图 1.6.8)。

```
defects = ['pits  ';'cracks ';'holes ';'dents '];
```

```
quantity = [5,3,19,25];
pareto(quantity,defects)
```

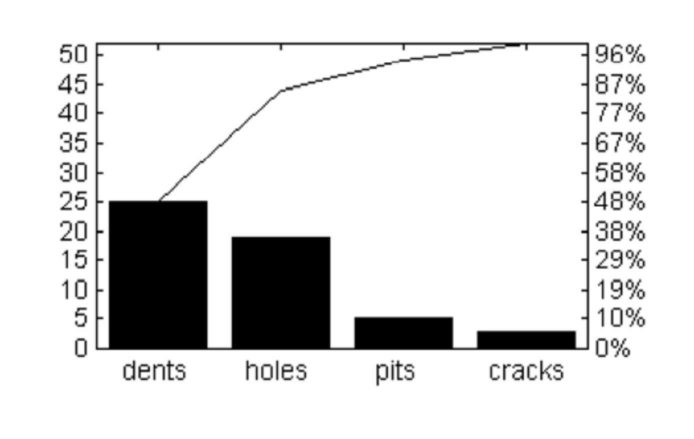


图 1.6.8　一批零件的帕累托图

参见：bar, capaplot, ewmaplot, hist, histfit, schart, xbarplot。

1.6.10　分位数-分位数图

- qqplot

功能：两个样本的分位数-分位数图。

格式：qqplot(X,Y)

qqplot(X,Y,pvec)

h = qqplot(...)

说明：qqplot(X,Y)显示两个样本的分位数-分位数图(参见图 1.6.9)。如果两个样本来源于同一分布，那么，图中的曲线为直线。若 X 和 Y 为矩阵，则为它们的每列数据绘制单独的曲线。图中样本数据以“＋”符号表示。并将位于第一分位数和第三分位数间的数据拟合绘制成一条线(这是两个样本顺序统计量的鲁棒线性拟合)。此线外推至样本数据的两端，以帮助用户评估数据的线性程度。

qqplot(X,Y,pvec)函数可在 pvec 矢量中规定分位数。

h = qqplot(...)则返回线段的句柄。

举例：首先产生两个不同均值和标准差的正态分布样本，然后作出其分位数-分位数图(见图 1.6.9)。

```
x=normrnd(0,1,100,1)
y=normrnd(0.5,2,50,1)
qqplot(x,y)
```

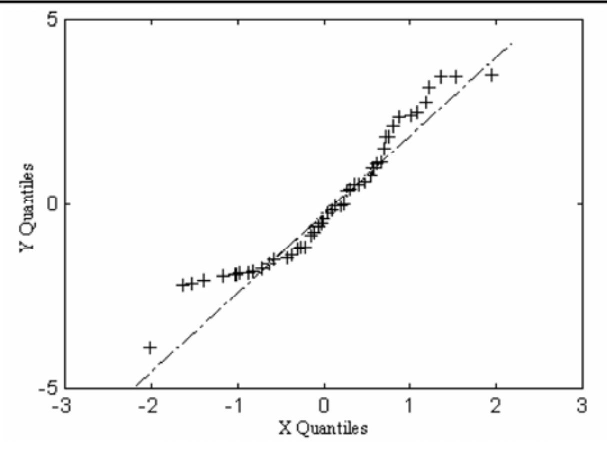


图 1.6.9 两个正态分布样本的分位数-分位数图

1.6.11 回归残差图

- rcoplot

功能：回归残差图。

格式：rcoplot(r,rint)

说明：rcoplot(r,rint)将试验样本回归后的残差及其置信区间绘制成误差条图(可参见图 1.6.10)，其中 r 和 rint 来自函数 regress 的输出。图中按数据的顺序给出各数据点的误差条。

举例：
```
X = [ones(10,1) (1:10)'];
y = X*[10;1] + normrnd(0,0.1,10,1);
[b,bint,r,rint] = regress(y,X,0.05);
rcoplot(r,rint);
```

参见：regress。

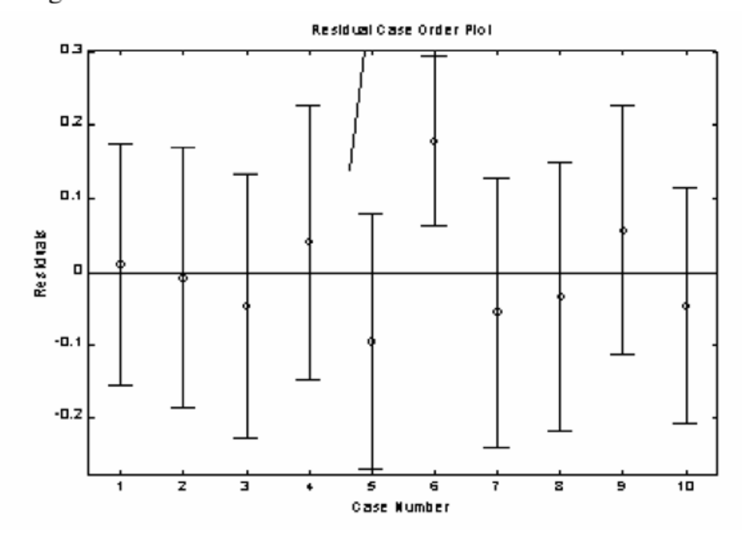


图 1.6.10 试验样本回归的回归残差图

1.6.12 参考多项式

- `refcurve`

功能：在当前图形中给出多项式拟合曲线。

格式：h = refcurve(p)

说明：refcurve(p)在当前图中给出多项式 p(以系数矢量 p 表示)的曲线(参见图 1.6.11)。n 阶多项式函数为

$$y = p_1x^n + p_2x^{n-1} + \Lambda + p_nx + p_{n-1} \tag{1.6.1}$$

h = refcurve(p)返回曲线的句柄。

举例：绘制火箭飞行高度图，即用多项式表达其理论高度（假设无空气阻力），与实际飞行高度（图中以“＋”表示）进行比较。

$$h = -4.9x^2 + 100x + 0 \tag{1.6.2}$$

即初始高度为 0，速度为 100m/s。

```
h = [85 162 230 289 339 381 413 437 452 458 456 ?
440 400 356];
plot(h,'+')
refcurve([-4.9 100 0])
```

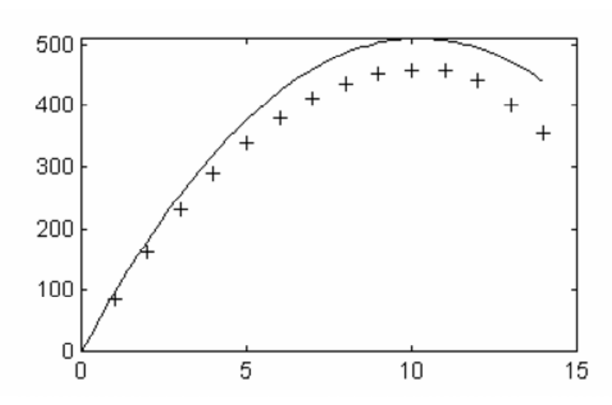


图 1.6.11 火箭飞行高度多项式拟合曲线

参见：polyfit, polyval, refline。

1.6.13 参考线

- `refline`

功能：在当前图中给出参考线。

格式：refline(slope,intercept)

refline(slope)

h = refline(slope,intercept)

refline

说明：refline(slope,intercept)在图中给出斜率为 slope、截距为 intercept 的直线（参见图 1.6.12）。而 refline(slope)中，slope 为一个二元矢量，所给出的直线为

y= slope(2)+ slope(1) x (1.6.3)

h = refline(slope,intercept)返回直线的句柄。

refline 则给出图中各线性对象的最小二乘拟合线（以下线型除外：‘－’，‘－－’，‘.–’），这与函数 lsline 是相同的。

举例：
```
y = [3.2 2.6 3.1 3.4 2.4 2.9 3.0 3.3 3.2 2.1 2.6]';
plot(y,'+')
refline(0,3)
```

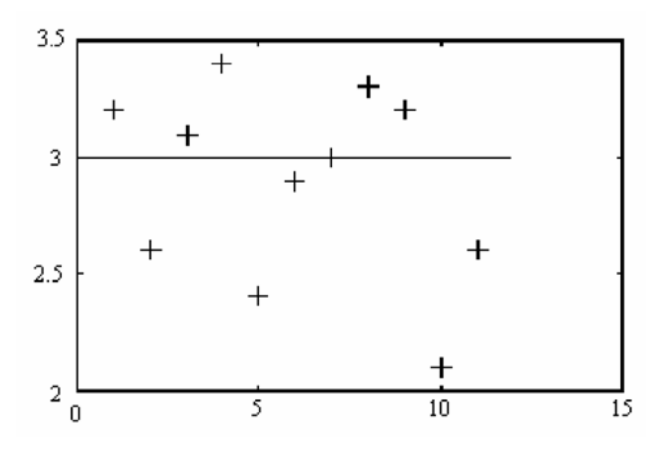


图 1.6.12 数据样本的参考线图

参见：lsline, polyfit, polyval, refcurve。

1.6.14 交互内插轮廓图

- surfht

功能：交互内插轮廓图。

格式：surfht(x,y,Z)

说明：在 surfht(x,y,Z)中，x 和 y 是曲线图中 x 轴和 y 轴的坐标矢量。x 的长度必须与 Z 的列数相同，y 的长度必须与 Z 的行数相同。surfht(x,y,Z)根据给出的 x、y、Z 数据，提供任意的 x、y 的坐标值上 Z 的内插值。图中给出了水平和垂直参考线，其交点确定当前的 x 坐标和 y 坐标。可以任意拖拉参考线，或者在 x 轴和 y 轴相应的编辑框内输入坐标值，则同时能够直接观察到更新的 Z 的内插值。

1.6.15　威布尔图

- weibplot

功能：威布尔分布概率图。

格式：weibplot(X)

h = weibplot(X)

说明：weibplot(X)将 X 中的数据显示成概率曲线(参见图 1.6.13)。如果 X 是矩阵，weibplot(X)则将 X 的每列各显示一条曲线。

h = weibplot(X)返回曲线的句柄。

威布尔分布概率图用于以图形方式来判断 X 中的数据是否来自威布尔分布。如果数据服从威布尔分布，则其曲线为直线；否则有一定的弯曲度。

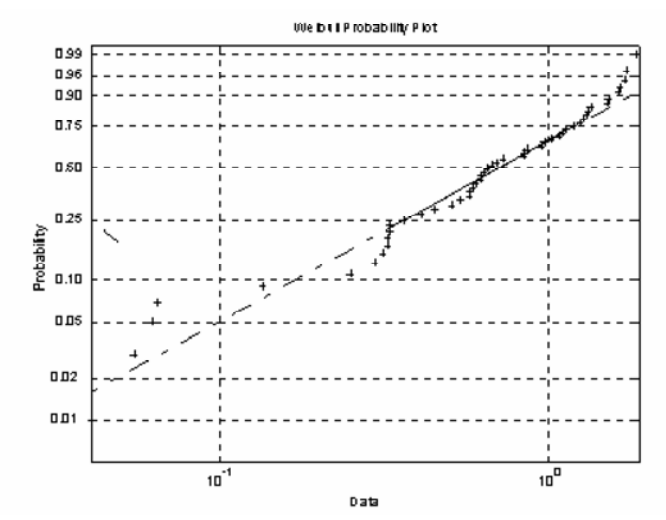


图 1.6.13　威布尔分布数据样本的概率图

举例：

```
r=weibrnd(1.2,1.5,50,1);
weibplot(r)
```

参见：normplot。

1.7　线 性 模 型

1.7.1　概述

线性模型是数理统计中的一个重要的应用分支，例如方差分析、回归分析等，都可归为这种模型。

线性模型通常可以表示为以下形式

$$y = x\beta + \varepsilon \qquad (1.7.1)$$

其中，y ——$n \times 1$观测值向量，

x ——模型的系数矩阵，

β —— $p \times 1$参数向量，

ε ——$n \times 1$随机干扰矢量。

线性统计模型的内容非常丰富，统计工具箱提供了最常用的一些功能。在方差分析方面，给出了单因素方差分析、双因素方差分析函数；在回归分析方面，给出了多重线性回归、多项式回归和逐步回归等函数；工具箱还提供了图形化的二次响应曲面模型。本章重点介绍上述函数的使用。

工具箱所提供的线性模型分析的主要函数见表 1.7.1。

表 1.7.1 线性模型函数表

函数分类	函数名称	功 能
方差分析	anova1	单因素方差分析
	anova2	双因素方差分析
回归分析	regress	多重线性回归
	lscov	给定方差矩阵回归
	ridge	岭回归
	stepwise	逐步回归 GUI
多项式回归	polyfit	多项式拟合
	polyval	多项式预测
	polyconf	给出置信区间的多项式预测
二次响应曲面模型	rstool	二次响应曲面模型的交互式图形工具

1.7.2 方差分析

方差分析法，因其基于统计数据的总变动中因素引起的变动成分与随机误差成分的比较而得名，是基于不多的统计数据，定量地分析一个或多个因素对某个应变量的影响和作用的显著性。其前提条件，是在各因素作用下应变量分布的正态性和等方差性。

(1) 单因素方差分析

单因素方差分析的基本问题，是比较和估计多个等方差正态总体的均值。其基本模型如下：

$$y_{ij} = \alpha_{oj} + \varepsilon_{ij} \qquad (1.7.2)$$

其中，y_{ij} ——观测值矩阵，

a_{oj} ——其各列为组均值的矩阵(a_{oj}表示 a 中第 j 列的所有行)，

ε_{ij} ——随机扰动矩阵。

标准的 ANOVA 表具有 4 列：平方和 SS、自由度 df、均方值（SS/df）、F 统计量。可以使用 F 统计量来进行假设检验，判断样本的各列数据是否为同一均值。单因素方差分析函数 anova1 的使用说明如下。

- anova1

  功能：单因素方差分析。

格式：p = anova1(X)

p = anova1(x,group)

说明：anova1(X)对样本 X 中的两列或多列数据进行均衡的单因素方差分析，以比较各列的均值。函数返回“零假设”（即 X 中各列的均值相同）成立的概率值。如果概率值接近于零，则零假设值得怀疑，表明各列的均值事实上是不同的。

anova1(x,group) 对样本 x 中由矢量 group 索引的两组或多组数据进行单因素方差分析以比较各列的均值。输入参数 group 标明矢量 x 中相应元素的组别。group 中的值为整数，最大值为需要比较的不同组的数量，最小值为 1。每组中至少应有一个元素，但并不要求每组中元素个数相同，因此适合于数据不均衡的情况。用以决定结果是否具有统计上的显著性的概率值大小限制的选择留给用户。

anova1 同时还显示一张表和一幅图。表为标准的 ANOVA 表（参见表 1.7.2），表中将 X 中数据的变化分成两部分：

- 由各列均值差异而产生的变化，
- 由各列的数据及其均值间的差异而导致的变化。

ANOVA 表具有 5 列数据：

第一列标明数据源；

第二列给出数据源的均方和（SS）；

第三列给出相应数据源的自由度 df；

第四列给出均方值，即比率 SS/df；

第五列给出 F 统计量。

p 值是 F 的函数(fcdf)。随着 F 的增加，p 值减少。图 1.7.1 为显示 X 中每列数据的 box 图。box 图中,各列数据的图的中心线若表现出较大差异，则相应于 F 值较大以及 p 值较小。

举例：X 中的五列数据分别为 1～5 的常数与均值为 0、标准差为 1 的正态随机干扰量之和。

```
x = meshgrid(1:5)
x =
  1 2 3 4 5
  1 2 3 4 5
  1 2 3 4 5
  1 2 3 4 5
x = x + normrnd(0,1,5,5)
x =
  2.1650 3.6961 1.5538 3.6400 4.9551
  1.6268 2.0591 2.2988 3.8644 4.2011
  1.0751 3.7971 4.2460 2.6507 4.2348
  1.3516 2.2641 2.3610 2.7296 5.8617
```

```
  0.3035 2.8717 3.5774 4.9846 4.9438
p = anova1(x)
p =
  5.9952e-05
```

由计算结果可知，观察所提供的 X 的随机数据样本，给出其各列均值相同结论的概率小于十万分之六。

表 1.7.2 ANOVA 表

Source	SS	df	MS	F
Columns	32.93	4	8.232	11.26
Error	14.62	20	0.7312	
Total	47.56	24		

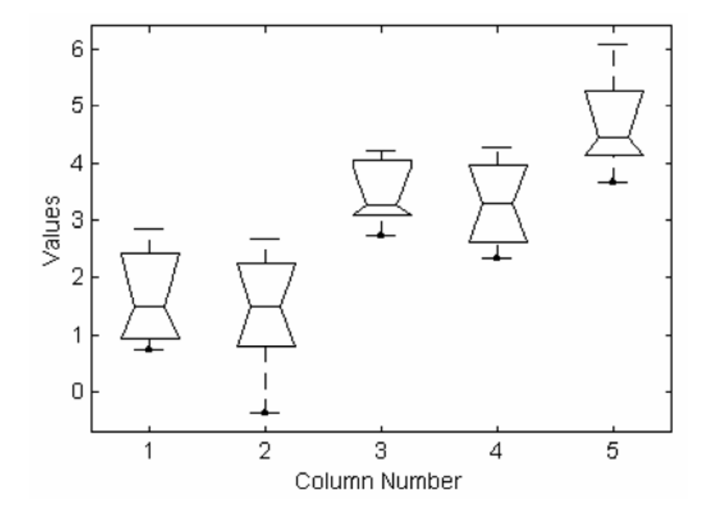


图 1.7.1 单因素方差分析的 box 图

下面的例子源自结构横梁材料的应力研究。在每千分之一英尺材料上施以 3000 磅力，观察横梁的挠曲，观测结果保存在矢量 strength 中。强韧的横梁挠曲较少。建筑工程师进行此项研究的目的在于，确定钢质横梁的应力强度是否等同于其他两种更昂贵的合金的应力强度。在矢量 alloy 中，钢材编码为 1，其他两种合金材料的编码分别为 2 和 3。结果见表 1.7.3 和图 1.7.2。

```
strength = [82 86 79 83 84 85 86 87 74 82 78 75 76 77 79 ...
79 77 78 82 79];
alloy =[1 1 1 1 1 1 1 1 2 2 2 2 2 2 3 3 3 3 3 3];
p = anova1(strength,alloy)
p =
   1.5264e-04
```

表 1.7.3 横梁应力强度的 ANOVA 表

Source	SS	df	MS	F
Columns	184.8	2	92.4	15.4
Error	102	17	6	
Total	286.8	19		

结果表明，三种材料的应力强度是明显不同的。Box 图显示出钢材的挠曲比其他两种合金的挠曲程度更大。

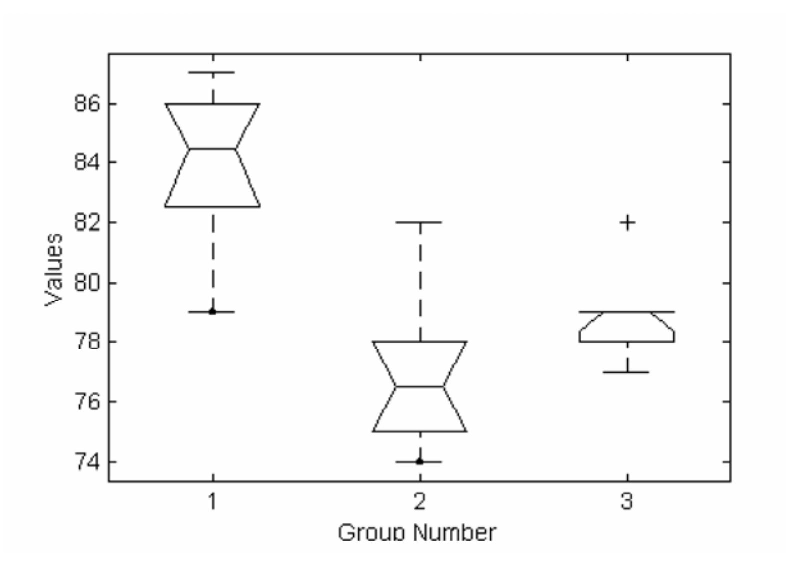


图 1.7.2　横梁应力强度的 box 图

参考：Hogg, R. V. and J. Ledolter. Engineering Statistics. MacMillan Publishing Company, 1987。

(2)　双因素方差分析

双因素方差分析是一种两因素、多水平析因试验数据的统计分析方法，目的在于确认来自不同组的数据是否具有相同的均值。

假设一个汽车制造公司有两个厂，都分别制造 3 种汽车。我们来考察汽车的燃气里程（即每升汽油所跑的里程数）随汽车种类不同变化和随工厂不同而变化的情况。由于工厂制造方法的差异，使燃气里程有总体的差别；而由于设计规定的差异，不同类汽车的燃气里程也可能不同。另外，制造方法和设计规定二者也可能存在综合效应，而影响汽车的燃气里程。因此，除非对工厂和汽车种类相结合进行观察，否则不可能观测到交互作用。

双因素方差分析是处理这种问题的典型方法。

首先建立问题的数学模型

$$y_{ijk} = \mu + \alpha_{oj} + \beta_{io} + \gamma_{ij} + \varepsilon_{ijk} \qquad (1.7.3)$$

其中

y_{ijk} ——观测值矩阵，

u ——样本总均值(常数均值)，

α_{oj} ——列元素为组均值的矩阵(各行 α 总和为 0)，

β_{io} ——行元素为组均值的矩阵(各列 β 总和为 0)，

$\gamma_{i,j}$ ——交互作用项(矩阵)(各行列 γ 的总和为 0)，

ε_{ijk} ——随机干扰矩阵。

调用汽车模型数据和双因素方差分析函数（详细使用说明见下文），来观察其结果：

```
load mileage
mileage
mileage =
   33.3000 34.5000 37.4000
   33.4000 34.8000 36.8000
   32.9000 33.8000 37.6000
   32.6000 33.4000 36.6000
   32.5000 33.7000 37.0000
   33.0000 33.9000 36.7000
cars = 3;
p = anova2(mileage,cars)
p =
   0.0000  0.0039  0.8411
```

mileage 数据给出，共有三种汽车(3 列元素分别代表不同的车种的燃气里程)和两个工厂(两行元素分别表示不同工厂生产的车的燃气里程)。mileage 有 6 行数据，是因为每个工厂为每种汽车提供了 3 辆车的测试数据(前三行数据属于第一工厂，后三行数据属于第二工厂)。

anova2 函数给出的标准方差分析(ANOVA)表(见表 1.7.4)。

表 1.7.4　　横梁应力强度的 ANOVA 表

Source	SS	df	MS	F
Columns	53.35	2	26.68	234.2
Rows	1.445	1	1.445	12.69
Interaction	0.04	2	0.02	0.1756
Error	1.367	12	0.1139	
Total	56.2	17		

用 F 统计量来假设检验里程数与车种、工厂以及车种-工厂交互作用的依赖关系。anova2 则直接给出了检验的 p 值。结果表明，车种的 p 值为 0.0000，明显表示里程数随车种的变化而变化；工厂的 p 值为 0.0039，表明里程数也因工厂的不同而存在差异。而车种-工厂交互作用项的 p 值为 0.8411，表明交互作用项几乎不起作用。

注意，anova2 所返回的 p 值依赖于对模型方程中随机干扰量所作的假设。要获得正确的 p 值，要求干扰量是独立正态分布，且方差为常数。

- anova2

  功能：双因素方差分析。

  格式：p = anova2(X,reps)

  说明：anova2(X,reps)进行均衡的双因素方差分析，比较数据样本 X 中多行和多列数据的均值。不同列中的数据表示单一因素的变化情况；不同行中的数据表示另一因素的变化情况。如果每种行-列对(“单元”)有不止一个的观测值，则用参数 reps 来标明每个“单元”多个观测值的不同标号。下面的矩阵中，

列因素有两种水平；行因素有三种水平，但每种水平有两组样本，相应地用脚标来标识。

$$\begin{bmatrix} x_{111} & x_{121} \\ x_{112} & x_{122} \\ x_{211} & x_{221} \\ x_{212} & x_{222} \\ x_{311} & x_{321} \\ x_{312} & x_{322} \end{bmatrix}$$

anova2 返回“零假设”(即列数据的均值与行数据的均值相同)成立的概率值 p。如果概率值接近于零，则零假设值得怀疑。用以决定结果是否具有统计上的显著性的概率值限制的选择留给用户。通常认为，如果 p 值小于 0.05 或 0.1，则结果较显著。

anova2 同时也显示一个标准方差分析表(ANOVA 表)，其中，按照 reps 参数值将 X 中数据的变化情况分成三或四部分：

• 由各列均值差异而产生的变化，
• 由各行均值差异而产生的变化，
• 由列和行因素的交互作用而导致的变化(如果 reps 值大于其缺省值 1),
• 其他因素。

ANOVA 表共有 5 列数据：

• 第一列标明数据源，
• 第二列给出数据源的均方和（SS），
• 第三列给出相应数据源的自由度 df，
• 第四列给出均方值，即比率 SS/df，
• 第五列给出 F 统计量，即均方比。

p 值是 F 的函数（fcdf）。随着 F 的增加，p 值减少。

1.7.3　回归分析

实际中，回归分析就是用统计数据寻求变量间关系的近似表达式——经验公式，并利用所得公式进行统计描述、分析和推断，解决预测、控制和优化问题。

(1)　多重线性回归

多重线性回归亦称“多元线性回归”、“复线性回归”。其回归函数关于未知参数和回归变量都是线性的多元回归，其变量关系为

$$Y=\beta_0+\beta_1 X_1+\Lambda+\beta_m X_m+e \qquad (1.7.4)$$

其中，β_i(i=0,1,...,m)是回归系数，e 是随机误差,Ee=0,De=σ^2。

公式(1.7.4)的另一种表达方式为：

$$y = X\beta + \varepsilon \tag{1.7.5}$$

$$\varepsilon \sim N(0, \sigma^2 I)$$

其中，y—— $n*1$ 观测值矢量，

x—— $n*p$ 回归矩阵，

β—— $p*1$ 参数矢量，

ε—— $n*1$ 随机干扰矢量。

多重线性回归的主要内容是：

1) 从对 X_1, ..., X_m, Y 的观测数据出发，求出回归系数 β_0, β_1, . . . , β_m 的估计量

$$\hat{\beta}_0, \hat{\beta}_1, \Lambda\ \hat{\beta}_m$$

2) 检验回归方程及各回归系数的显著性。

3) 利用经验回归方程

$$\hat{Y} = \hat{\beta}_0 + \hat{\beta}_1 X_1 + \Lambda + \hat{\beta}_m X_m \text{(即 } \hat{y} = X\hat{\beta} + \varepsilon \text{)}$$

进行预测和控制。

回归方程（1.7.5）的最小二乘解为

$$b = \hat{\beta} = (X'X)^{-1} X'y \tag{1.7.6}$$

多重线性回归函数 regress 用 QR 分解法(即矩阵的正交三角分解)计算 b。计算 b 并不一定用 QR 分解法，但矩阵 R 却可用来计算置信区间。

观测数据的预测值可按下述公式计算

$$\hat{y} = Xb = Hy \quad , H = X(X'X)^{-1} X' \tag{1.7.7}$$

残差 r 为 y 的观测值和预测值之差

$$r = y - \hat{y} = (I - H) y \tag{1.7.8}$$

在模型方程中，残差对应于误差项 ε，因此残差可用于检验模型假设方面的错误。因为线性回归的一个前提假设是 ε 服从均值为 0、方差为常数的独立正态分布。

然而，残差之间却是相关的，而且其方差大小依赖于观测点的位置。因此常常对残差进行“学生化”处理,意为使各残差具备相同的方差。

在下面的方程中，“学生化”的残差 t_i，服从自由度为$(n-p)$的学生 t 分布。

$$t_i = \frac{r_i}{\hat{\sigma}_{(i)} \sqrt{1 - h_i}}$$

其中

$$\hat{\sigma}^2_{(i)} = \frac{\|r\|^2}{n - p - 1} - \frac{r_i^2}{(n - p - 1)(1 - h_i)} \tag{1.7.9}$$

t_i —— 第 i 个数据点的“学生化” 残差

r_i —— 第 i 个数据点的初始残差

n —— 样本容量

p —— 模型中参数的数量

h_i —— 矩阵 H 的第 i 个对角元素

$\hat{\sigma}^2_{(i)}$ ——除去第 i 个数据点后误差（error）的方差的估计

通过比较 t_i 和学生 t 分布的临界值，可以对野值进行假设检验。如果 t_i 很大，可以怀疑观测值为野值。

每一残差均值的置信区间为

$$c_i = r_i \pm t_{\left(1-\frac{\alpha}{2},\nu\right)} \hat{\sigma}_{(i)} \sqrt{1-h_i} \tag{1.7.10}$$

如果置信区间中未包含 0，则意味着在显著性概率为 α 的条件下，残差均值为 0 的假设是不能接受的。

- `regress`

功能：多重线性回归。

格式：b = regress(y,X)

[b,bint,r,rint,stats] = regress(y,X)

[b,bint,r,rint,stats] = regress(y,X,alpha)

说明：b = regress(y,X)返回基于观测值 y 和回归矩阵 X 的最小二乘拟合 β 的结果。

[b,bint,r,rint,stats] = regress(y,X)则给出 β 的估计 b、β 的 95%置信区间（p*2 矢量 rint）、残差 r 以及每个残差的 95%置信区间（n*2 矢量 rint）；矢量 stats 给出回归的 R^2 统计量和 F 以及 p 值。

[b,bint,r,rint,stats] = regress(y,X,alpha)的不同点在于它给出的结果置信区间为 100(1－alpha)%，其他都相同。

举例：假设真实模型为

$$y = 10 + x + \varepsilon \tag{1.7.11}$$

$$\varepsilon \sim N(0, 0.01I)$$

其中 I 为单位矩阵。

```
X = [ones(10,1) (1:10)']
X =
   1   1
   1   2
   1   3
   1   4
   1   5
   1   6
   1   7
```

```
    1   8
    1   9
    1   10
y = X*[10;1] + normrnd(0,0.1,10,1)
y =
    11.1165
    12.0627
    13.0075
    14.0352
    14.9303
    16.1696
    17.0059
    18.1797
    19.0264
    20.0872
[b,bint] = regress(y,X,0.05)
b =
    10.0456
    1.0030
bint =
    9.9165 10.1747
    0.9822 1.0238
```

参考：Chatterjee, S. and A. S. Hadi. Influential Observations, High Leverage Points, and Outliers in Linear Regression. Statistical Science, 1986. pp. 379－416.

(2) 给定方差矩阵的线性回归

lscov 函数提供了给定方差时线性模型的求解方法，以下是函数的使用说明。

- lscov

功能：给定方差矩阵的线性回归。

格式：X = lscov (A,B,V)

[X, DX] = lscov (A,B,V)

说明：X = lscov (A,B,V)按最小二乘法求解回归方程 A*X = B + E，返回回归系数 X。其中，E 服从正态分布，均值为 0，方差为 V；A 为 M×N 矩阵，且 M > N；经典的线性代数求解公式为

$$X = inv(A'*inv(V)*A)*A'*inv(V)*B \tag{1.7.12}$$

而用最小二乘法得到的结果为

$$X = min[(A*X-B)'*inv(V)*(A*X-B)] \tag{1.7.13}$$

[X, DX] = lscov (A,B,V)除返回回归系数 X 外，还返回系数的标准误差 DX。

DX 的标准计算公式为：

mse = B'*(inv(V) −inv(V)*A*inv(A'*inv(V)*A)*A'*inv(V))*B./(m−n) (1.7.14)

dx = sqrt(diag(inv(A'*inv(V)*A)*mse)) (1.7.15)

参见：slash, nnls, qr。

(3) 岭回归

岭回归亦称“脊回归估计”、“岭估计”，是一种改进最小二乘估计的方法，适用于线性回归模型中正规方程的系数矩阵 $X'X$ 接近奇异（即$|X'X|$的值较小）时的情形。

- ridge

功能：岭回归的参数估计。

格式：b = ridge(y,X,k)

说明：b = ridge(y,X,k)返回岭回归系数 b。

求解所基于的线性模型为

$$y = X\beta + \varepsilon \tag{1.7.16}$$

其中，X——$n*p$ 矩阵

y ——$n*1$ 观测矢量

k ——标量常数（岭系数）

β 的岭估计为

$$b = (X'X + kI)^{-1}X'y \tag{1.7.17}$$

当 k=0，b 为最小二乘估计。增加 k 值，b 的偏度增加，但方差减小。应用中需要根据具体问题选择合适的 k 值来权衡岭估计 b 的偏倚和均方误差，通常取 0<k<1。

举例：本例显示了随 k 值增加，岭系数的变化情况(见图 1.7.3)。数据采用 hald 数据库。

```
load hald;
b = zeros(4,100);
kvec = 0.01:0.01:1;
count = 0;
for k = 0.01:0.01:1
count = count + 1;
b(:,count) = ridge(heat,ingredients,k);
end
plot(kvec',b'),xlabel('k'),ylabel('b',?
'FontName','Symbol')
```

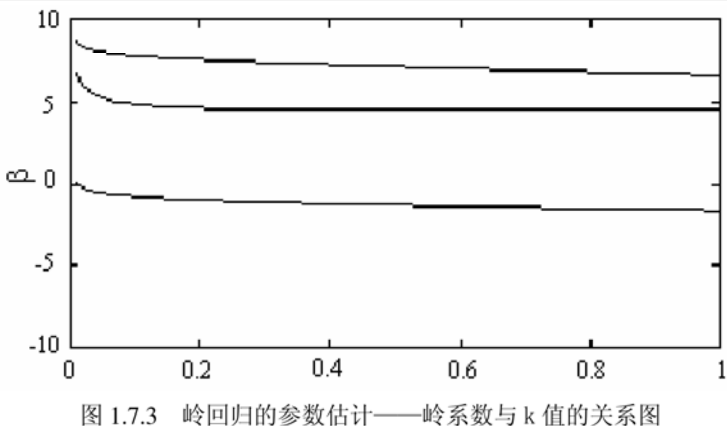


图 1.7.3　岭回归的参数估计——岭系数与 k 值的关系图

参见：regress,stepwise。

(4)　逐步回归的交互式图形环境

逐步回归是选择回归变量，产生最优多重回归方程的一种常用数学方法。其基本思想是，将回归变量一个一个地选入，选入的条件是其偏回归平方和显著；每选入一个新变量后，对已入选的各变量逐个进行显著性检验，并剔除不显著变量。如此反复选入、检验、剔除，直至无法剔除且无法选入为止。

逐步回归也可分为两种方法，一种是前向逐步回归，即开始时模型中无任何变量(模型项)，每一步选入待选项中一个显著性最高的项(具有高 F 统计量和低 p 值的变量)，直至无可选项为止；另一种是后向逐步回归，首先将所有待选项纳入模型中，然后剔除最不显著的变量，直至剩余的变量都是显著项为止。

多重线性回归中一个经常遇到的问题是输入变量间具有交互作用而影响到输出量，即存在相关性。在这种情况下，模型中一个输入变量可能会“屏蔽”其他变量对结果的“贡献”。作为一个固定运算过程，逐步回归函数对使用者来说是稍具风险的，因为回归的结果是与初始模型和选入策略密切相关的。

统计工具箱提供了具备交互式图形用户界面(GUI)的逐步回归工具 stepwise 函数，使用户能够更直观地理解和操纵回归模型。用户可用 hald 数据库来研究 stepwise GUI，具体可以如下开始：

```
load hald
stepwise(ingredients,heat)
```

其中，hald 数据库源自不同的水泥混合物反应热的研究。每种混合物中有 4 种成分，反应所产生的热量取决于混合物中各成分的量的多少。

- **stepwise**

  功能：逐步回归的交互式环境。

  格式：stepwise(X,y)

  　　　stepwise(X,y,inmodel)

  　　　stepwise(X,y,inmodel,alpha)

说明：逐步回归的交互式界面由三个相互联系的图形窗口组成：

- 逐步回归图
- 逐步回归诊断表
- 逐步回归历史图

每个窗口都有激活区。当鼠标移动至其中一个区域时，由“箭头”变成“圆圈”，点击此点则可在此区域进行下一步操作。

stepwise(X,y)将给定的数据 X 和 y 进行逐步回归拟合。显示三个交互式图形窗口，以控制回归条件和移动模型项。

stepwise(X,y,inmodel)可以控制初始回归模型项。矢量 inmodel 的值，用来标明包含在初始模型中的矩阵 X 的列号。

stepwise(X,y,inmodel,alpha)可以进行拟合系数的置信区间的控制。alpha 为测试模型中每项显著性的指标。缺省地，$alpha=1-(1-0.025)^{(1/p)}$,其中，p 为矩阵 X 的列数。

最小二乘系数由填充绿色的圆绘出。如果某系数的置信区间穿越白色的值为 0 的线，那么此系数显然与 0 相差不多。影响显著的模型项用实线绘出，明显接近于 0 的项则用点线给出。

点击图中列表的回归系数及其置信区间项，可以改变其当前状态。如果某项包含在回归模型内，则此项用绿色给出；若排除在模型外，则用红线给出。

参考：Draper, Norman and Smith, Harry, Applied Regression Analysis,Second Edition, John Wiley & Sons, Inc. 1981 pp. 307－312。

参见：regstats, regress, rstool。

1.7.4　多项式回归

多项式回归是指回归函数是回归变量的多项式的回归。由于任一函数均可用多项式逼近，因此多项式回归有广泛的应用。下面是工具箱所提供的多项式拟合函数(polyfit)、多项式评估(polyval)、多项式预测(polyconf)和拟合多项式及预测的交互式绘图函数(polytool)的简要说明。

1　polyfit

功能：多项式曲线拟合。

格式：[p,S] = polyfit(x,y,n)

说明：p=polyfit(x,y,n)是在最小二乘意义上，将数据 y(i)=p(x(i))拟合成次数为 n 的多项式

$$p(x) = p_1x^n + p_2x^{n-1} + \text{K} + p_nx + p_{n+1} \tag{1.7.18}$$

结果以系数矢量 p 的形式给出。p 为一行矢量，其长度为 $n+1$，给出按降次幂排列的多项式的系数。

[p,S] = polyfit(x,y,n)返回多项式系数 p 和矩阵 S。S 用于 polyval 函数，可进行

预测的误差估计。如果 Y 中数据的误差服从方差为常值的独立正态分布，则 polyval 产生至少包含预测的 50%的误差界。

如果无意将 S 传递给 polyval 或 polyconf 来计算误差的估计值，则可略去 S。

举例：

```
[p,S] = polyfit(1:10,[1:10] + normrnd(0,1,1,10),1)
p =
     1.0300 0.4561
S =
   -19.6214  -2.8031
     0        -1.4639
     8.0000    0
     2.3180    0
```

参见：polyval, polytool, polyconf。

2 polyval

功能：多项式评估。

格式：Y = polyval(p,X)

[Y,DELTA] = polyval(p,X,S)

说明：Y = polyval(p,X)返回给定系数 p 和变量 x 值的多项式的预测值。

[Y,DELTA] = polyval(p,X,S)使用 polyfit 函数的可选输出 S 来进行误差估计，给出 Y±DELTA。如果 polyfit 函数的输入数据 Y 的误差服从方差为常值的独立正态分布，则 polyval 产生的 Y±DELTA 至少包含预测值的 50%。如果 p 是矢量，其元素为按降次幂排列的多项式的系数，则 polyval(p,X)为 X 处多项式的预测值。如果 X 为矩阵或矢量，给出 X 的每点处(即取每个元素值)的多项式预测值。

举例：模拟函数 $y = x$，加入标准差为 0.1 的正态随机误差。用 polyfit 估计多项式的系数。

```
[p,S] = polyfit(1:10,(1:10) + normrnd(0,0.1,1,10),1);
X = magic(3);
[Y,D] = polyval(p,X,S)
Y =
   8.0696 1.0486 6.0636
   3.0546 5.0606 7.0666
   4.0576 9.0726 2.0516
D =
   0.0889 0.0951 0.0861
   0.0889 0.0861 0.0870
   0.0870 0.0916 0.0916
```

参见：polyfit, polytool, polyconf。

3　polyconf

功能：多项式预测和置信区间评估。

格式：[Y,DELTA] = polyconf(p,X,S)

[Y,DELTA] = polyconf(p,X,S,alpha)

说明：[Y,DELTA] = polyconf(p,X,S) 使用 polyfit 函数的可选输出 S 来给出 95%置信区间 Y±DELTA，假定 polyfit 输入数据的误差服从方差为常值的独立正态分布。

[Y,DELTA] = polyconf(p,X,S,alpha)则给出 100（1－alpha ）%置信区间。如果 alpha＝0.1，则获得 90%置信区间。如果 p 是矢量，其元素为按降次幂排列的多项式的系数（如 polyfit 的输出），则 polyconf (p,X)为 X 处多项式的预测值。如果 X 为矩阵或矢量，给出 X 的每点处（即取每个元素值）的多项式预测值。

举例：对列数为 100～200 的方阵的因式分解的计算时间进行预测并给出 90%置信区间。

```
n = [100 100:20:200];
for i = n
A = rand(i,i);
tic
B = lu(A);
t(ceil((I－80)/20)) = toc;
end
[p,S] = polyfit(n(2:7),t,3);
[time,delta_t] = polyconf(p,n(2:7),S,0.1)
time =
0.0829 0.1476 0.2277 0.3375 0.4912 0.7032
delta_t =
0.0064 0.0057 0.0055 0.0055 0.0057 0.0064
```

4　polytool

功能：拟合多项式及预测的交互式绘图。

格式：polytool(x,y)

polytool(x,y,n)

polytool(x,y,n,alpha)

说明：polytool 函数采用回归模型，通过最小二乘法进行拟合。

$$y_i = \beta_0 + \beta_1 x_i + \beta_2 x_i^2 + \Lambda + \beta_n x_i^n + \varepsilon_i \tag{1.7.19}$$

$$\varepsilon_i \sim N(0,\sigma^2) \qquad \forall i$$

$$Cov(\varepsilon_i, \varepsilon_j) = 0 \qquad \forall i, j$$

polytool(x,y)将列矢量 x 和 y 拟合为线，并显示其交互式图形。此图形为 GUI，可以通过改变所拟合多项式的阶次来研究拟合效果。图形同时对于曲线的新预测值显示所拟合的曲线，并给出全局 95%的置信区间。Y 的当前预测值及其不确定度也显示在 y 轴的左侧。

polytool(x,y,n)拟合初始阶次为 n 的多项式。

polytool(x,y,n,alpha)则同时绘出预测值的 100（1-alpha）%置信区间。参数 n 控制多项式拟合的阶次，可在图形上部的弹出式菜单中选择多项式的阶次。在 x 轴下的编辑框内键入 x 的值或用鼠标拖拉 x 的垂直标志线，或者用鼠标点击图中任一点时，x 的垂直标志线随即自动移至此处，编辑框内的 x 值也相应改变为此点处的值，图形中随即给出更新的 y 的预测值及其置信区间。当鼠标的箭头靠近 x 的垂直标志线时变为十字叉状，指示此时垂直标志线是可拖拉的。

1.7.5　二次响应曲面工具

响应曲面方法(Response Surface Methodology，RSM)用于理解多个输入变量和单个输出变量间的量的关系。

例如，单输出变量 z，是两个变量 x 和 y 的多项式函数。$z=f(x,y)$描述了三维空间(x,y,z)中的二维曲面。当然，也可以有多个输入变量，结果则为超曲面。对于三个输入(x_1,y_1,z_1)的情况，其二次响应曲面方程为

$$y = b_0 + b_1x_1 + b_2x_2 + b_3x_3 + \cdots \qquad \text{(线性项)}$$

$$b_{12}x_1x_2 + b_{13}x_1x_3 + b_{23}x_2x_3 + \cdots \qquad \text{（交互项）}$$

$$b_{11}x_1^2 + b_{22}x_2^2 + b_{33}x_3^2 \qquad \text{（二次项）}$$

统计工具箱所提供的二次响应曲面工具为 rstool 函数，说明如下。

- rstool

  功能：二次响应曲面的交互式拟合和可视化。

  格式：rstool(x,y)

  rstool(x,y,'model')

  rstool(x,y,'model',alpha,'xname','yname')

  说明：rstool(x,y)显示模型的交互式预测图，包括预测的 95%置信区间。图形源自运用线性可加模型对数据（x，y）的多重回归。

  rstool(x,y,'model')增加了对初始回归模型的控制。‘model’可为下列字符串：

  - ‘interaction’—— 包括常数项、线性项和交叉乘积项；

- ‘quadratic’—— 在‘interaction’的内容上再增加平方项；
- ‘purequadratic’—— 包括常数项、线性项和平方项。

rstool(x,y,'model',alpha)的区别在于用两条红色曲线给出预测值 100(1−alpha)%全局置信区间。如，当 alpha＝0.01 时，给出 99%置信区间。

rstool 显示图形的“矢量”，一个是输入矩阵 X 的每列数据矢量。响应变量 y 是与 X 的行数相匹配的列矢量。

rstool(x,y,'model',alpha,'xname','yname')可在图中的 X 轴和 Y 轴分别标上自定的‘xname’和‘yname’名称。

拖拉白色参考线(虚线)可以同时观察更新的预测值。或在参数 X 的编辑框内键入参数值，同样可以获得相应的预测值。

参见：nlintool。

1.8　非线性回归模型

1.8.1　概述

非线性回归模型(Nonliner Regression Models)是其回归函数关于未知参数具有非线性结构的回归模型。模型的拟合一般很困难，通常首先需猜未知参数的初始值，然后反复迭代。每次迭代都会修正当前的估测值，直至算法收敛为止。

统计工具箱所提供的函数见表1.8.1，可拟合以下形式的非线性模型

$$y = F(x,\beta) + \varepsilon \qquad (1.8.1)$$

其中 y——$n*1$观测矢量，

F——关于 x、β的函数，

x——$n*p$ 输入变量矩阵，

β——$p*1$ 待估计的未知参数矢量，

ε——$n*1$ 随机干扰矢量。

表 1.8.1　非线性回归函数表

函数名称	功　能
nlinfit	非线性最小二乘拟合
nlintool	非线性拟合预测图
nlparci	参数置信区间
nlpredci	预测置信区间
nnls	非负最小二乘拟合

1.8.2 非线性建模示例

以反应动力学中的Hougen-Watson模型为例。模型形式为

$$rate = \frac{\beta_1 \cdot x_2 - x_3 / \beta_5}{1 + \beta_2 \cdot x_1 + \beta_3 \cdot x_2 + \beta_4 \cdot x_3} \tag{1.8.2}$$

其中，β_1、β_2、...、β_5为未知参数，x_1、x_2、x_3为3个输入变量。三项输入为氢、n-戊烷和iso-戊烷。很显然，上述参数模型为非线性模型。

文件reaction.mat提供了反应的仿真数据。

```
load reaction
who
Your variables are:
      beta          rate          xn
      model     reactants     yn
```

各变量的意义为：

- rate ——13×1 反应率观测矢量；
- reactants ——13×3 反应物矩阵，即参量 x；
- beta ——5×1 参数的初始估计矢量，即参量β；
- model——非线性函数的名称；
- xn——反应物名称的字符串矩阵；
- yn——包含反应结果名称的串。

(1) Hougen-Watson模型拟合

统计工具箱提供了nlinfit函数来获取非线性模型的参数估计量，返回的参数值为其最小二乘估计。所采用的算法为全局收敛的Levenberg-Marquardt修正的高斯-牛顿法。

nlinfit函数的执行所需的参量首先是一种特定的非线性模型（以函数名称标志，拟合过程中通过此名称对函数进行调用），其他输入数据为相应的反应物数据、反应结果数据以及未知参数的初始估测值。在Matlab中，这样的函数称为函数的函数。

‘hougen’函数的定义可为如下形式：

```
function yhat = hougen(beta,x)
b1 = beta(1);
b2 = beta(2);
b3 = beta(3);
b4 = beta(4);
b5 = beta(5);
x1 = x(:,1);
x2 = x(:,2);
x3 = x(:,3);
```

```
yhat = (b1*x2 - x3/b5)./(1+b2*x1+b3*x2+b4*x3);
```

调用nlinfit函数对反应数据进行拟合。

```
load reaction
betahat = nlinfit(reactants,rate,'hougen',beta)
betahat =
1.2526
0.0628
0.0400
0.1124
1.1914
```

注意，‘hougen’函数名称所代表的函数即为(1.8.1)式。nlinfit函数还有两个可选的输出，即结果的残差和雅可比矩阵。残差为拟合值与观测值之差；雅可比矩阵则类似于标准线性回归模型中矩阵X。这两个可选的输出可用于参数估计值和反应结果预测值的置信区间计算。

(2)　参数估计的置信区间

使用函数nlparci，可以获得上述参数估计值的95%置信区间。

```
[betahat,f,J] = nlinfit(reactants,rate,'hougen',beta);
betaci = nlparci(betahat,f,J)
betaci =
 -1.7467      3.2519
 -0.0377      0.1632
 -0.0312      0.1113
 -0.0609      0.2857
 -1.7381      3.1208
```

(3)　结果预测值的置信区间

使用函数nlpredci，可以获得反应结果预测值的95%置信区间。

```
[yhat, delta] = nlpredci('hougen',reactants,betahat,f,J);
opd = [rate yhat delta]
opd =
  8.5500   8.2937    0.9178
  3.7900   3.8584    0.7244
  4.8200   4.7950    0.8267
  0.0200    -0.0725  0.4775
  2.7500   2.5687    0.4987
  14.3900  14.2227   0.9666
  2.5400   2.4393    0.9247
  4.3500   3.9360    0.7327
  13.0000  12.9440   0.7210
```

```
8.5000    8.2670     0.9459
0.0500   —0.1437    0.9537
11.3200  11.3484    0.9228
3.1300    3.3145     0.8418
```

其中，矩阵opd中的第一列为观测的反应率(rate)结果；第二列为预测值，其95％置信区间为第二列数据±第三列数据。

(4) 非线性拟合和预测的交互式图形用户界面

用于非线性模型的nlintool函数与多项式模型中的rstool类同，其输入参数与nlinfit的输入相同，它直接调用nlinfit。

nlintool 的功用不仅仅为单纯的拟合和预测，其交互式图形用户界面提供了反映多维非线性函数的变化规律的图形化综合环境。

1.8.3 非线性回归函数详解

1 nlinfit

功能：用高斯-牛顿方法进行非线性最小二乘拟合。

格式：[beta,r,J] = nlinfit(X,y,'model',beta0)

说明：beta = nlinfit(X,y,'model',beta0)返回由‘model’描述的非线性函数的系数。‘model’是用户提供的函数，其形式为

$$\hat{y} = f(\beta, X)$$

即在给定独立变量 X 和参数估计量 β 后，‘model’返回 y 的预测值。矩阵 X 中，每个独立变量有一列数据。响应结果 y 为一列矢量，其元素个数等于 X 的行数。

[beta,r,J] = nlinfit(X,y,'model',beta0)返回拟合系数 beta、残差 r 和雅可比矩阵 J，可提供给函数 nlintool 进行预测的误差估计。

举例：

```
load reaction
betafit = nlinfit(reactants,rate,'hougen',beta)
betafit =
             1.1323
             0.0582
             0.0354
             0.1025
             1.2801
```

参见：nlintool。

2 nlintool

功能：拟合非线性方程并绘制交互式图形。

格式：nlintool(x,y,'model',beta0)

nlintool(x,y,'model',beta0,alpha)

nlintool(x,y,'model',beta0,alpha,'xname','yname')

说明：nlintool(x,y,'model',beta0)提供对数据（x，y）进行非线性拟合的预测图形。其中，用两条红色曲线给出95%全局置信区间。beta0是包含参数的初始估测值的矢量。

nlintool(x,y,'model',beta0,alpha)的区别在于其给出预测值的 100(1-alpha)%置信区间。

nlintool 显示图形的“矢量”。一个是输入矩阵 X 的每列数据矢量。响应变量 y 是与 X 的行数相匹配的列矢量。

alpha 的缺省值为 0.05，即对应 95%置信区间。

nlintool(x,y,'model',beta0,alpha,'xname','yname')可以在图中的 X 轴和 Y 轴分别标上自定的‘xname’和‘yname’名称。

拖拉白色参考线（虚线）可同时观察更新的预测值。或在参数 X 的编辑框内键入参数值，同样可以获得相应的预测值。使用“Model”的菜单可交互地改变模型；使用“Export”菜单可将特定的变量移至基本工作台。

参见：nlinfit, rstool。

3　nlparci

功能：非线性模型的参数估计的置信区间。

格式：ci = nlparci(beta,r,J)

说明：nlparci(beta,r,J)是在给定残差 r 和雅可比矩阵 J 条件下，给出 beta 参数的最小二乘估计的 95%置信区间。仅对于其雅可比矩阵 J 的行数大于 beta 的长度的系统，置信区间的计算才是有效的。

nlparci 使用 nlinfit 的输出值作为其输入值。

举例：继续 nlinfit 函数详解中的举例。

```
load reaction
[beta,resids,J]=nlinfit(reactants,rate,'hougen',?
 beta);
ci = nlparci(beta,resids,J)
ci =
      －1.0798 3.3445
      －0.0524 0.1689
      －0.0437 0.1145
      －0.0891 0.2941
      －1.1719 3.7321
```

参见：nlinfit, nlintool, nlpredci。

4　nlpredci

功能：非线性模型预测的置信区间。

格式：ypred = nlpredci('model',inputs,beta,r,J)

[ypred,delta] = nlpredci('model',inputs,beta,r,J)

说明：ypred = nlpredci('model',inputs,beta,r,J)在给定参数β、残差 r 和雅可比矩阵 J 的条件下，返回预测的非线性系统响应值 ypred。输入 inputs 是非线性函数中独立变量值的矩阵。

[ypred,delta] = nlpredci('model',inputs,beta,r,J)则增加了非线性最小二乘预测的 95％置信区间 delta。仅当 r 的长度超过 beta 的长度以及 J 是列满秩时，置信区间的计算才是有效的。

nlpredci 将 nlinfit 的输出作为其输入。

举例：继续 nlinfit 函数详解中的举例。

```
load reaction
[beta,resids,J]=nlinfit(reactants,rate,'hougen',?
beta);
ci = nlpredci('hougen',reactants,beta,resids,J)
ci =
        8.2937
        3.8584
        4.7950
      －0.0725
        2.5687
        14.2227
        2.4393
        3.9360
        12.9440
        8.2670
      －0.1437
        11.3484
        3.3145
```

参见：nlinfit, nlintool, nlparci。

5 nnls

功能：非负最小二乘拟合。

格式：X = nnls (A,b)

[X,W] = nnls (A,b)

说明：X = nnls (A,b)返回使 norm(A*X – b)最小的大于或等于零的矢量 X。A、b 必须为实数。当 X 中的元素小于 0 时，则用缺省容差 TOL 确定，其中，TOL =max(size(A)) * norm(A,1) * eps。另外，也可用 X = nnls (A,b,TOL)来替代。

[X,W] = nnls (A,b)还返回二重矢量 W。当 $x(i)$=0 时，w(i)<0;当 $x(i)$>0 时，$w(i)$

近似为零。

参见：lscov , slash。

1.9　多元统计分析

1.9.1　概述

多元分析，是基于同时对多个对象或变量联合观测所得的多元数据的分析，研究多个变量或多个属性表征的多元总体。

多元分析的主要方法，有主成分分析、聚类分析、典型相关分析，以及回归分析、方差分析和协方差分析等等。而主成分分析可以说是最常用的一种方法，统计工具箱所提供的多元统计方法即为主成分分析，其主要函数见表 1.9.1。

主成分分析亦称“主分量分析”或“分量分析”等，是将多个相关变量简化为少数几个不相关变量的一种多元统计方法。其目的在于简化统计数据和揭示变量间的关系。每个主成分是初始变量的线性组合，所有的主成分间相互正交，所以没有冗余信息，它们构成数据空间的正交基。从数学的角度看，其根本思想在于降维。而其降维是从简化方差和协方差的结构来考虑的。

主成分的全体决定于变量正交集的维数。但通常情况下，前几个主成分的方差之和会超过初始数据方差总和的 80%。

多元统计的固有困难之一是多维变量的可视化问题。在 Matlab 中，plot 命令能够显示二维变量关系图形；plot3 和 surf 命令能够显示不同的三维视图；当变量超过 3 个时，则需通过想象来把握它们之间的关系。

表 1.9.1　多元统计分析函数表

函数名称	功　　能
princomp	初始数据矩阵的主成分分析
pcacov	运用协方差矩阵进行主成分分析
pcares	主成分分析残差
barttest	巴特力特检验

1.9.2　主成分分析函数详解

1　princomp

功能：主成分分析。

格式：PC = princomp(X)

[PC,SCORE,latent,tsquare] = princomp(X)

说明：[PC,SCORE,latent,tsquare] = princomp(X)对数据矩阵 X 进行主成分分析，给出各主成分（PC）、所谓的 Z-得分(SCORE)、X 的方差矩阵的特征值(latent)和

每个数据点的 Hotelling T^2 统计量(tsquare)。

举例：计算 Hald 数据库中组成成分数据的主成分以及每个成分的方差。

```
load hald;
[pc,score,latent,tsquare] = princomp(ingredients);
pc,latent
pc =
 0.0678  -0.6460   0.5673  -0.5062
 0.6785  -0.0200  -0.5440  -0.4933
 -0.0290   0.7553   0.4036  -0.5156
 -0.7309  -0.1085  -0.4684  -0.4844
latent =
 517.7969
 67.4964
 12.4054
 0.2372
```

参考：J. Edward Jackson, A User's Guide to Principal Components, John Wiley & Sons, Inc. 1991. pp. 1-05。

参见：barttest, pcacov, pcares。

2 pcacov

功能：运用协方差矩阵进行主成分分析。

格式：pc = pcacov(X)

[pc,latent,explained] = pcacov(X)

说明：[pc,latent,explained] = pcacov(X)通过协方差矩阵 X 进行主成分分析，返回主成分(pc)、协方差矩阵 X 的特征值(latent)和每个特征向量表征在观测量总方差中所占的百分数（explained）。

举例：

```
load hald
covx = cov(ingredients);
[pc,variances,explained] = pcacov(covx)
pc =
  0.0678 -0.6460  0.5673 -0.5062
  0.6785 -0.0200 -0.5440 -0.4933
 -0.0290  0.7553   0.4036 -0.5156
 -0.7309 -0.1085 -0.4684 -0.4844
variances =
  517.7969
  67.4964
  12.4054
   0.2372
```

```
explained =
   86.5974
   11.2882
   2.0747
   0.0397
```

参考：J. Edward Jackson, A User's Guide to Principal Components, John Wiley & Sons, Inc. 1991. pp. 1-25。

参见：barttest, pcares, princomp。

3 pcares

功能：主成分分析的残差。

格式：residuals = pcares(X,ndim)

说明：pcares(X,ndim)返回保留 X 的 ndim 个主成分所获的残差。注意，ndim 是一个标量，必须小于 X 的列数。而且，X 是数据矩阵，而不是协方差矩阵。

举例：本例表明随主成分维数从 1 增加到 3，hald 数据的第一行的残差不断下降。

```
load hald
r1 = pcares(ingredients,1);
r2 = pcares(ingredients,2);
r3 = pcares(ingredients,3);
r11 = r1(1,:)
r11 =
  2.0350     2.8304     -6.8378   3.0879
r21 = r2(1,:)
r21 =
  -2.4037    2.6930     -1.6482    2.3425
r31 = r3(1,:)
r31 =
 0.2008   0.1957     0.2045     0.1921
```

参考：J. Edward Jackson, A User's Guide to Principal Components, John Wiley & Sons, Inc. 1991. pp. 1-25。

参见：barttest, pcacov, princomp。

4 barttest

功能：主成分的巴特力特检验。

格式：ndim = barttest(x,alpha)

[ndim,prob,chisquare] = barttest(x,alpha)

说明：巴特力特检验是一种等方差性检验。ndim = barttest(x,alpha)是在显著性水平 alpha 下，给出满足数据矩阵 X 的非随机变量的 n 维模型，ndim 即模型维数，它由一系列假设检验所确定。ndim＝1 表明数据 X 对应于每个主成分的方差

是相同的；ndim＝2表明数据X对应于第二成分及其余成分的方差是相同的，等等。

[ndim,prob,chisquare] = barttest(x,alpha)则返回模型维数、假设检验的显著性值和检验的 x^2 值。

举例：
```
x = mvnrnd([0 0], [1 0.99; 0.99 1],20);
x(:,3:4) = mvnrnd([0 0], [1 0.99; 0.99 1],20);
x(:,5:6) = mvnrnd([0 0], [1 0.99; 0.99 1],20);
[ndim, prob] = barttest(x,0.05)
ndim =
3
prob =
0
0
0
0.5081
0.6618
1.0000
```

参见：princomp, pcacov, pcares。

1.9.3 主成分分析示例

让我们来看一个应用的例子，其中采用了329个美国城市的生活质量的9种不同指数的数据："气候"、"住房"、"健康"、"犯罪"、"交通"、"教育"、"艺术"、"娱乐"和"经济"。对于这些指标，其数值较高的表示为较好的，如"犯罪"的较高指数意味着低的犯罪率。

首先从cities.mat中调入数据。

```
load cities
whos
   Name Size
   categories 9 by 14
   names 329 by 43
   ratings 329 by 9
```

城市数据库中包含三类变量：

- 类别(categories) —— 为指标名的字符串，
- 名称(names) —— 329个城市的字符串矩阵，
- 比率(ratings) —— 329×9的指标数据矩阵。

如类别(categories)的内容为：

```
categories
categories =
```

```
climate
housing
health
crime
transportation
education
arts
recreation
economics
```

绘制 Box 图(见图 1.9.1)，看一下指标数据 ratings 的大致情况。

```
boxplot(ratings,0,'+',0)
set(gca,'YTicklabels',categories)
```

从图中可以注意到，“艺术”和“住房”的变化比“犯罪”和“气候”的变化要大得多。

当所有变量具有相同的单位时，计算初始数据的主成分，其意义是明显的。但当各指标数据单位不同，或不同指标的数据变化差异很大时，需要对数据进行标准化处理。可用每个指标数据(即每列数据)的标准差去除相应列的数据元素。

```
stdr = std(ratings);
sr = ratings./stdr(ones(329,1),:);
```

然后，就可开始寻找主成分。

```
[pcs, newdata, variances, t2] = princomp(sr);
```

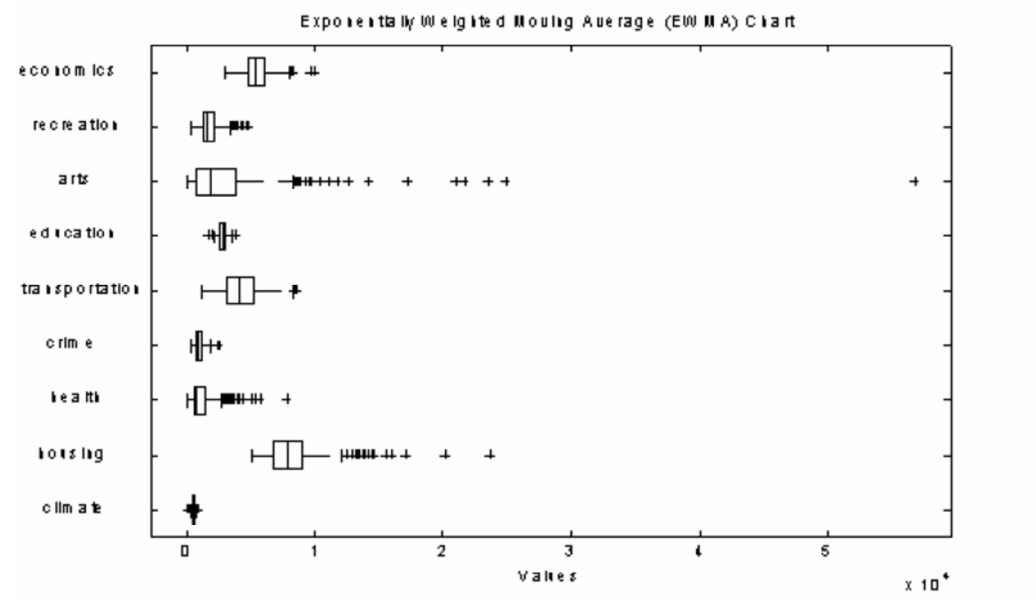


图 1.9.1　9 种生活质量指数(ratings)的 Box 图

(1)　主成分(第一个输出量)

princomp 函数的第一个输出量 pcs，是 9 个主成分。这是初始变量的线性组合。

可以看一下前三个主成分矢量。

```
 p3 = pcs(:,1:3)
p3 =
    0.2064    -0.2178      0.6900
    0.3565    -0.2506      0.2082
    0.4602     0.2995      0.0073
    0.2813    -0.3553     -0.1851
    0.3512     0.1796     -0.1464
    0.2753     0.4834     -0.2297
    0.4631     0.1948      0.0265
    0.3279    -0.3845      0.0509
    0.1354    -0.4713     -0.6073
```

结果表明，第一列(即第一主成分)中的最大的权重为第 3 个元素(对应于“艺术”)和第 7 个元素(对应于“健康”)。 第一主成分中的所有元素符号相同，是所有变量的加权平均。

为表现主成分间的正交性，用主成分矩阵 p3 的转置矩阵与其相乘，可得单位矩阵。

```
I = p3'*p3
I =
1.0000      0.0000    －0.0000
 0.0000      1.0000    －0.0000
－0.0000   －0.0000      1.0000
```

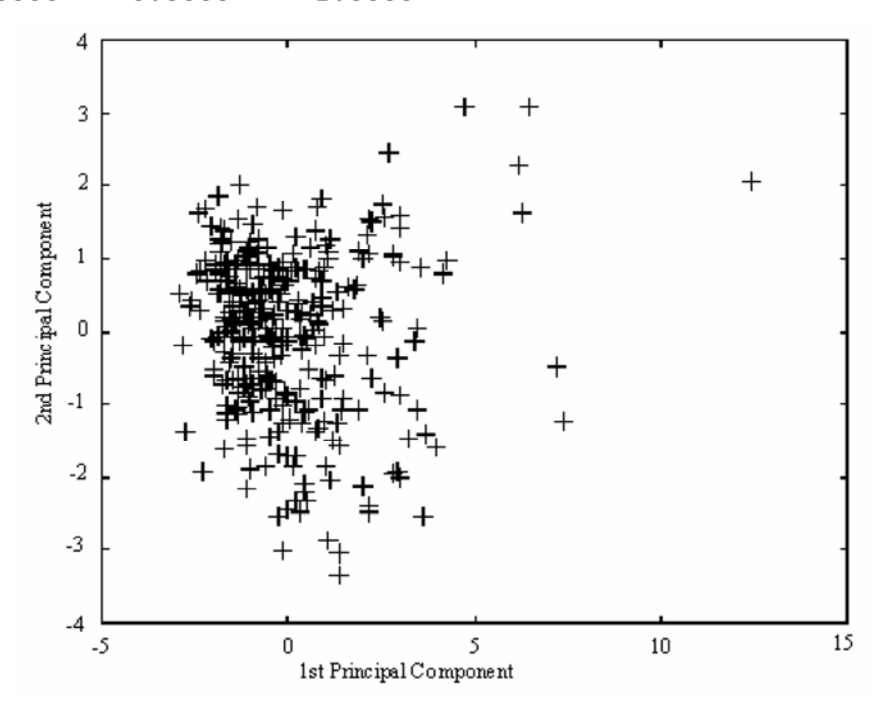


图 1.9.2 主成分得分图

(2) 主成分得分(第二个输出量)

princomp 函数的第二个输出量 newdata，是初始数据在主成分所定义的新坐标系中的坐

标，它与输入矩阵的大小相同。

newdata 的前两列数据显示原比率(ratings)数据投影在第一和第二主成分上的结果(见图 1.9.2)。

```
plot(newdata(:,1),newdata(:,2),'+')
xlabel('1st Principal Component');
ylabel('2nd Principal Component');
```

此时，可用 gname 函数来标明图中的点的名称，如

```
gname(names)
```

然后可用鼠标点击需要标记的数据点，最后按下“return”键，得到图 1.9.3。

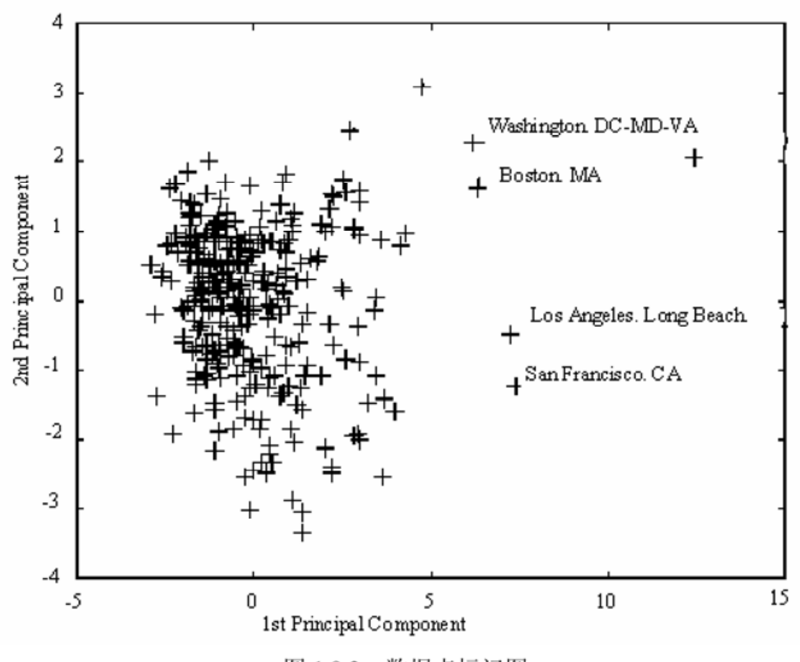


图 1.9.3　数据点标记图

(3)　方差(第三个输出量)

princomp 函数的第三个输出量 variance，是 newdata 矩阵各列数据相应的方差。

```
variances
variances =
    3.4083
    1.2140
    1.1415
    0.9209
    0.7533
    0.6306
    0.4930
    0.3180
    0.1204
```

用 pareto 图可以直观地显示各主成分方差所占的比例(见图 1.9.4)：

```
percent_ explained = 100*variances/sum(variances)
pareto(percent_explained)
xlabel('Principal Component')
ylabel('Variance Explained (%)')
```

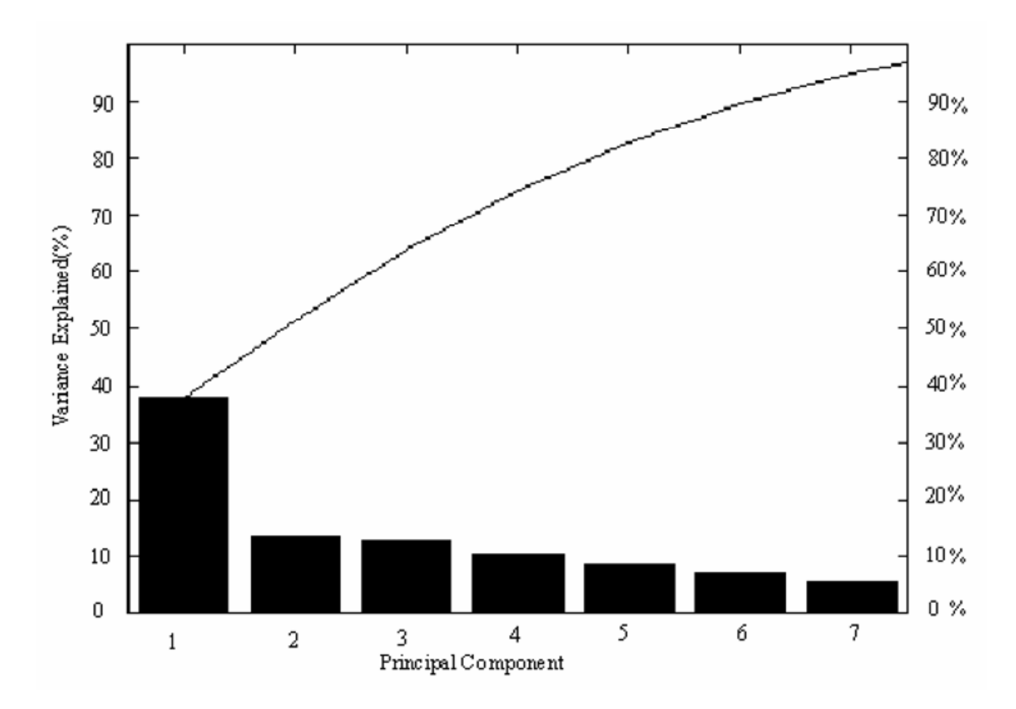


图 1.9.4　各主成分方差的 pareto 图

(4) Hotelling T^2统计量(第四个输出量)

princomp 函数的最后一个输出量是 Hotelling T^2统计量，它是每一个观测量与数据样本中心的距离的统计度量。以下过程是分析与找出数据中最偏远的点。

```
[st2, index] = sort(t2);   % 升阶搜索
st2 = flipud(st2);   %降阶值
index = flipud(index);   % 降阶索引
extreme = index(1)
extreme =
213
names(extreme,:)
ans =
New York, NY
```

1.10 试验设计

1.10.1 概述

数据与其中所含的信息是存在差别的。若要从数据中提取信息，则需对产生数据的系统作出一定的假设。运用这些假设和实际理论可以建立系统的数学模型。

通常，即便严格用公式表达的模型仍有未知的常量。试验的目的就是获得数据以估计这些常数。为什么要进行试验呢？可在系统运转过程中进行观测研究，这样迟早会获得我们需用的所有数据。事实上，这是相当通用的方法。而在统计建模中所关心的历史数据一般有如下三个特征。

- 观测系统中诸变量的变化，以及系统输出随之相应的变化情况。需要注意的是，系统的变化并不一定意味着其输出的变化。
- 统计建模中通常假设各观测量是相互独立的。但系统运转过程中，实际情况并不一定如此。
- 控制具体系统的运转常常是依次改变控制变量来观测其结果。倘若同时改变两个以上变量，则各变量对结果的影响是无法隔离的。

试验设计可以直接解决这些问题，其最大的优点是能够主动地操纵所研究的系统。它比被动地进行测量需要的数据量少，但获取信息的质量却较高。

统计工具箱提供了一些函数（见表 1.10.1），以针对不同的情况进行试验设计。

表 1.10.1　试验设计

函数名称	功　能
ff2n	二级全因素设计
fullfact	混合水平全因素设计
rowexch	行交换 D-优化设计
cordexch	使用坐标交换进行 D-优化设计
daugment	设计的 D-优化参数
dcovary	固定方差的 D-优化设计

1.10.2 试验设计基本过程

(1) 全析因设计

假设在产品的切削加工中，需要确定加工过程的不一致性是源自机床还是操作员。如果一个操作员总是操作同一台机器，则无法知道输出产品的差异性是由何种因素造成的。只有通过让每一个操作员分别轮换操作每一台机器，才可分离出各因素对结果的影响效果。这就是析因方法。

函数 fullfact 即为析因设计函数。

设有四个操作员和三台机床，下面为析因设计过程。

```
d = fullfact([4 3])
d =
             1     1
             2     1
             3     1
             4     1
             1     2
             2     2
             3     2
             4     2
             1     3
             2     3
             3     3
             4     3
```

d 的每一行表示不同操作员和不同机器的一种组合。本例中的组合有 12 种（即 d=12）。

析因设计的一种特殊情况为每种因子仅取两个值。假设需要很快决定 3 个变量因子取值分别为 0 和 1 时对结果的影响，则可如下进行试验设计：

```
d2 = ff2n(3)
d2 =
0    0    0
0    0    1
0    1    0
0    1    1
1    0    0
1    0    1
1    1    0
1    1    1
```

即可进行 2^3＝8 种组合试验。

(2) 部分析因设计

析因设计的困难之一是随析因变量的增加，试验组合数呈指数增长。在上例中，如果析因变量不是 3 个而是 7 个，则全析因试验次数达 2^7＝128 次。

假设系统中诸变量间无协同作用，试验次数可以大为减少。故常常采用部分析因设计。

(3) D-优化设计

全析因设计和部分析因设计用于 20 世纪早期。20 世纪 70 年代，统计学家开始运用计算机按照优化方法重新进行试验设计。D-优化设计是将 Fisher 信息矩阵 $X'X$ 的行列式最大化的方法之一，而(X′ X)正比于参数的协方差矩阵的逆矩阵，因此，最大化 det(X′ X)即等价于使参数的最小二乘估计具有最小方差。即 D-优化设计是使线性模型参数 β 的回归估计的

置信椭球的体积最小化的方法。

统计工具箱中提供的 D-优化设计的函数为 cordexch、daugment、dcovery 和 rowexch。

cordexch 和 rowexch 是在设定模型后进行 D-优化设计的两种性能相当的优化算法。它们都是迭代算法，通过递增其基本元素来不断改进其初始设计。cordexch 的坐标交换算法的增量是设计矩阵的单个元素；而 rowexch 的行交换算法的增量是设计矩阵的行。

进行 D-优化设计，首先需要规定输入的数量、试验的数量和所拟合模型的阶数。cordexch 和 rowexch 都用以下的字符串对模型进行规定：

- 'linear' ('l') ——缺省模型，具有常数项和一阶项；
- 'interaction' ('i') ——包括常数项、线性项和交叉乘积项；
- 'quadratic' ('q') —— ‘interaction’ 的各项加上平方项；
- 'purequadratic' ('p') ——包括常数项、线性项和平方项。

或者，可用一个整数矩阵来设定这些项。详情参见应用函数 x2fx 的说明。

示例：考虑具有两个因子变量的二次模型，使用坐标交换算法进行试验设计。模型的形式为

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_{12} x_1 x_2 + \beta_{11} x_1^{\ 2} + \beta_{22} x_2^{\ 2} + \varepsilon$$

假设用 D-优化设计方法进行 9 次试验来拟合此模型。

```
settings = cordexch(2,9,'q')
settings =
 -1        1
  1        1
  0        1
  1       -1
 -1       -1
  0       -1
  1        0
  0        0
 -1        0
```

绘制成图形为图 1.10.1。

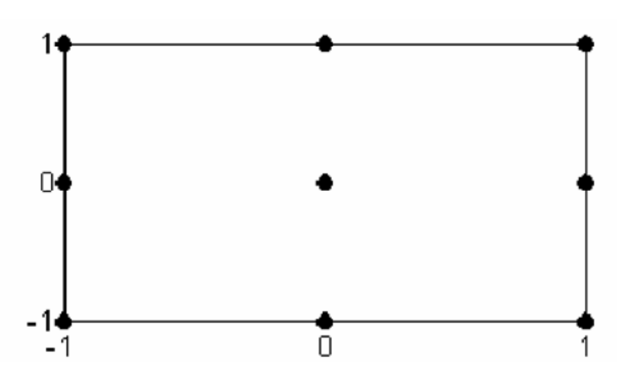


图 1.10.1　试验的 D-优化设计(坐标交换算法)方案

```
h = plot(settings(:,1),settings(:,2),'.');
set(gca,'Xtick',[-1 0 1])
set(gca,'Ytick',[-1 0 1])
set(h,'Markersize',20)
```

举例：考虑具有两个因子变量的交互作用模型，使用行交换算法进行试验设计。模型的形式为

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_{12} x_1 x_2 + \varepsilon$$

用 D-优化设计方法进行 4 次试验来拟合此模型。

```
[settings, X] = rowexch(2,4,'i')
settings =
    -1         1
    -1        -1
     1        -1
     1         1
X =
     1    -1     1    -1
     1    -1    -1     1
     1     1    -1    -1
     1     1     1     1
```

settings 矩阵给出如何改变试验输入量。X 为拟合上述试验回归模型的设计矩阵。X 的第一列用于拟合常数项，最后一列为第二列和第三列数据的乘积。其相应的图为图 1.10.2。

```
h = plot(settings(:,1),settings(:,2),'.');
set(gca,'Xtick',[-1 0 1])
set(gca,'Ytick',[-1 0 1])
set(h,'Markersize',20)
```

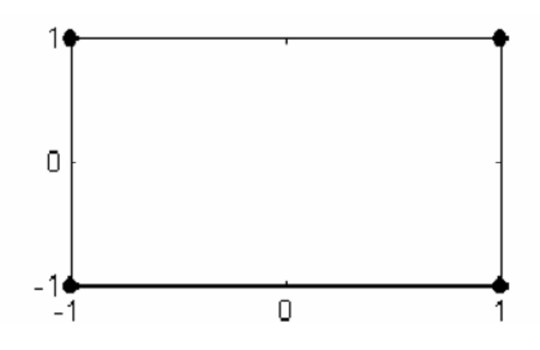


图 1.10.2 试验的 D-优化设计(行交换算法)方案

实际上，试验是一个迭代过程。为更多地了解系统模型，我们经常会进行全面的试验。函数 daugment 可供进行优化的额外试验。

假设对于具有 4 个变元的线性模型进行拟合，已经做了 8 次试验。

```
settings = cordexch(4,8)
settings =
   1    -1    1    1
  -1    -1    1   -1
  -1     1    1    1
   1     1    1   -1
  -1     1   -1    1
   1    -1   -1    1
  -1    -1   -1   -1
   1     1   -1   -1
```

这种试验设计对4个输入变量的线性模型拟合已经是足够了，但无法拟合出交叉乘积(即交互作用)项。如果我们想增加8次试验来获得这些项，可以如下进行。

```
[augmented, X] = daugment(settings,8,'i');
augmented
augmented =
     1   -1    1    1
    -1   -1    1   -1
    -1    1    1    1
     1    1    1   -1
    -1    1   -1    1
     1   -1   -1    1
    -1   -1   -1   -1
     1    1   -1   -1
    -1   -1   -1    1
     1    1    1    1
    -1   -1    1    1
    -1    1    1   -1
     1   -1    1   -1
     1   -1   -1   -1
    -1    1   -1   -1
     1    1   -1    1
info = X'*X
info =
   16 0 0 0 0 0 0 0 0 0 0
   0 16 0 0 0 0 0 0 0 0 0
   0 0 16 0 0 0 0 0 0 0 0
   0 0 0 16 0 0 0 0 0 0 0
   0 0 0 0 16 0 0 0 0 0 0
   0 0 0 0 0 16 0 0 0 0 0
```

```
0 0 0 0 0 0 16 0 0 0 0
0 0 0 0 0 0 0 16 0 0 0
0 0 0 0 0 0 0 0 16 0 0
0 0 0 0 0 0 0 0 0 16 0
0 0 0 0 0 0 0 0 0 0 16
```

1.10.3 实验设计函数详解

1 ff2n

功能：两种水平全因子设计。

格式：X = ff2n(n)

说明：X = ff2n(n)产生两种水平的设计试验矩阵 X。n 为因子数量，X 的列数为 n，行数为 2^n。

举例：

```
X = ff2n(3)
X =
 0    0    0
 0    0    1
 0    1    0
 0    1    1
 1    0    0
 1    0    1
 1    1    0
 1    1    1
```

可见，X 是 0～2^n-1 的二进制组合的全体。

参见：fullfact。

2 fullfact

功能：混合水平全因子试验设计。

格式：design = fullfact(levels)

说明：design = fullfact(levels)给出全析因试验的因子设置矩阵。矢量 levels 的每个元素给定试验方案设计矩阵中相应列的取值水平范围。例如，levels 的第一个元素值为 3，则试验方案设计矩阵中第一列的范围为 1～3 的整数。

举例：设 levels＝[2，4]，则 fullfact 设计出 8 次试验，第一因子(第一列)取 2 种水平，第二因子(第二列)取 4 种水平。

```
d = fullfact([2 4])
d =
1   1
2   1
```

```
1   2
2   2
1   3
2   3
1   4
2   4
```

参见：ff2n, dcovary, daugment, cordexch。

3 x2fx

功能：将因子设置矩阵转换成设计矩阵。

格式：D = x2fx(X)

D = x2fx(X,'model')

说明：x2fx 是 rstool、regstats 和 cordexch 等函数的工具函数。

D = x2fx(X)将系统输入矩阵 X 转换成带常数项的线性模型的设计矩阵。

D = x2fx(X,'model')则还可控制模型的阶数。'model' 可以是以下字符串：

- 'interaction'——包括常数项、线性项和交叉乘积项；
- 'quadratic'——在 'interaction' 基础上增添平方项；
- 'purequadratic'——包括常数项、线性项和平方项。

参数 'model' 也可为矩阵。在这种情况下，'model' 的每一行代表一项，而其各列的值对应于 X 的相应列的此项的指数，见举例(2)。因此，可以给出任意阶数的多项式模型。

举例：(1)

```
x = [1 2 3;4 5 6]'; model = 'quadratic';
D = x2fx(x,model)
D =
    1  1  4   4  1  16
    1  2  5  10  4  25
    1  3  6  18  9  36
```

设 x_1 为 x 的第一列，x_2 为第二列。则，D 矩阵的第一列为常数项；第二列为 x_1 项；第三列为 x_2 项；第四列为 $x_2 x_1$ 项；第五列为 x_1^2 项；第二列为 x_2^2 项。

(2)

```
x = [1 2 3]' ;
model = [0 1 2]';
D = [1 1 1; 1 2 4; 1 3 9]
```

D 的第一列对应为 x 的第一列的零阶项；D 的第一列和第三列分别对应为 x 的第一列的一阶项和二阶项。

参见：rstool, cordexch, rowexch, regstats。

4 cordexch

功能：使用坐标交换进行 D-优化设计。

格式：settings = cordexch(nfactors,nruns)

[settings,X] = cordexch(nfactors,nruns)

[settings,X] = cordexch(nfactors,nruns,'model')

说明：settings = cordexch(nfactors,nruns)产生因子试验设置矩阵 settings，其列数为 nfactors，行数为 nruns。对于 D-优化设计，使用附加常数项的线性模型。

[settings,X] = cordexch(nfactors,nruns)产生关联设计矩阵 X。

[settings,X] = cordexch(nfactors,nruns,'model')给出拟合特定回归模型的设计。输入参量'model'为以下字符串之一：

- 'interaction'——包括常数项、线性项和交叉乘积项；
- 'quadratic'——在'interaction'基础上增添平方项；
- 'purequadratic'——包括常数项、线性项和平方项。

举例：用二次模型进行 9 次双因子 D-优化设计试验方案如下：

```
settings = cordexch(2,9,'quadratic')
settings =
 -1      1
  1      1
  0      1
  1     -1
 -1     -1
  0     -1
  1      0
  0      0
 -1      0
```

参见：rowexch, daugment, dcovary, hadamard, fullfact, ff2n。

5 daugment

功能：设计的 D-优化参数。

格式：settings = daugment(startdes,nruns)

[settings,X] = daugment(startdes,nruns,'model')

说明：settings = daugment(startdes,nruns)是在初始设计的试验（startdes）基础上再给出 nruns 次新的试验因子设置。

[settings,X] = daugment(startdes,nruns,'model')提供设计矩阵 X。'model'的意义同前所述，控制模型的阶数，可选项为'interaction'、'quadratic'和'purequadratic'。

举例：对 2^2 因子设计，用增加 5 次设置的方案来拟合二次模型。

```
startdes = [-1 -1; 1 -1; -1 1; 1 1];
settings = daugment(startdes,5,'quadratic')
```

```
settings =
   -1 -1
    1 -1
   -1  1
    1  1
    1  0
   -1  0
    0  1
    0  0
    0 -1
```

结果为 3^2 因子设计。

参见：cordexch, dcovary, rowexch。

6　dcovary

功能：规定了固定方差的 D-优化设计。

格式：settings = dcovary(factors,covariates)

[settings,X] = dcovary(factors,covariates,'model')

说明：settings = dcovary(factors,covariates)给出满足于固定方差为 covariates 的 D-优化设计，factors 为所希望控制的变元数。

[settings,X] = dcovary(factors,covariates,'model')同时给出设计矩阵 X；'model'的意义和选项与前同。

举例：

```
covariates = dummyvar([1 1 2 2 3 3 4 4]);
settings = dcovary(2,covariates(:,1:3),'linear')
settings =
    1   1   1  0  0
   -1  -1   1  0  0
   -1   1   0  1  0
    1  -1   0  1  0
    1   1   0  0  1
   -1  -1   0  0  1
   -1   1   0  0  0
    1  -1   0  0  0
```

参见：daugment, cordexch。

1.11 统计工序管理图

1.11.1 概述

统计工序管理(Statistical Process Control,SPC)是指运用数理统计的一系列理论和方法，对生产过程或产品质量的管理和控制。各种抽样和验收方法，各种统计估计、比较和分析方法，产品质量和生产过程的管理图，以及其他各种描述统计图和分析图等都是统计质量管理常用的方法和工具，其核心是管理图。这种方法操作起来很简单，易于在生产过程中实施。统计工具箱提供了主要的一些统计质量管理图函数，见表 1.11.1。

表 1.11.1 工序质量管理图函数

函 数 名 称	功 能
xbarplot	均值时间曲线
schart	标准差的时间曲线
ewmaplot	指数加权滑动平均图
capable	工序性能指数
capaplot	工序性能图
histfit	直方图和正态密度曲线
normspec	绘制规定区间的正态密度曲线

1.11.2 管理图

管理图(control chart)是 1924 年由美国人休哈特(W.A.Shewhart)首创而得名。它是在质量管理过程中，记录受控特征或质量特征的一种图表，是统计质量管理的得力工具之一。

统计工具箱提供了三种通用的管理图：标准差管理图(S control char)、均值管理图($\overline{X}$ control chart)和指数加权滑动平均图(Exponentially Weighted Moving Average charts, EWMA)。

1 schart

功能：标准差管理图(S-图)。

格式：schart(DATA)

schart(DATA,conf)

schart(DATA,conf,specs)

[outliers,h] = schart(DATA,conf,specs)

说明：schart(DATA)绘制 DATA 中数据的标准差管理图，见图 11.1.1。DATA 中的每行数据为按时间顺序给出的观测值，上下管理限 UCL 和 LCL 为此工序中新观测值的 99%置信区间。

schart(DATA,conf)可用 conf 给出自定的置信区间管理限。如 conf＝0.95 时，图形则绘出 95%置信区间。

schart(DATA,conf,specs)可用两个元素矢量 specs 限制所绘曲线的界限。

[outliers,h] = schart(DATA,conf,specs)给出 DATA 中那些数据均值失控的行的序号(返回矢量 outliers 中)，另外还给出曲线的句柄(矢量 h)。

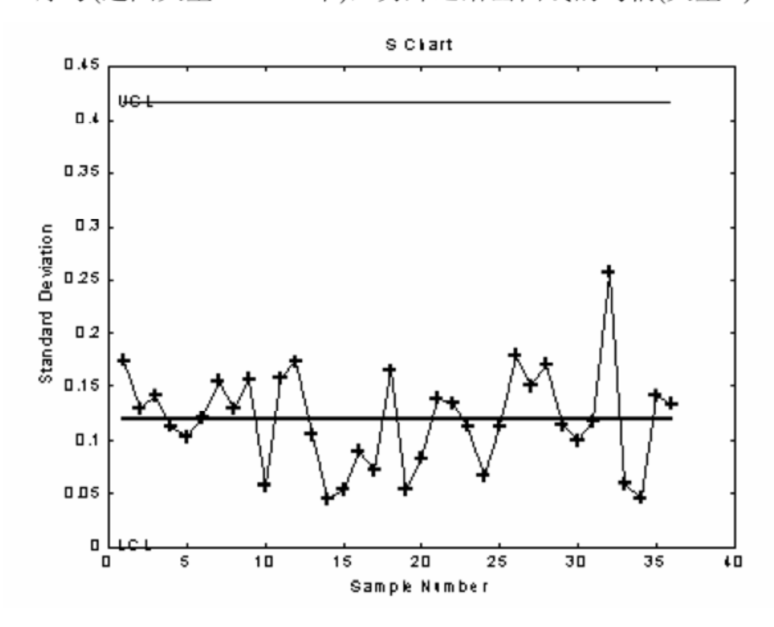


图 1.11.1　生产零件尺寸的标准差管理图

举例：本例给出了新生产的零件尺寸的标准差管理图(图 1.11.1)。生产过程中每小时测一次，共监测了 36 小时。矩阵 runout 中每一行包含 4 个随机抽取零件的尺寸。

```
load parts
schart(runout)
```

参考：Montgomery, Douglas, Introduction to Statistical Quality Control, John Wiley & Sons 1991. p. 235。

参见：capaplot, ewmaplot, histfit, xbarplot。

2　xbarplot

功能：均值管理图。

格式：xbarplot(DATA)

xbarplot(DATA,conf)

xbarplot(DATA,conf,specs)

[outlier,h] = xbarplot(...)

说明：xbarplot(DATA)给出 DATA 数据的均值管理图，见图 1.11.2。DATA 中的每行数据为按时间顺序给出的观测值，上下管理限 UCL 和 LCL 为此工序中新观测值的 99%置信区间。

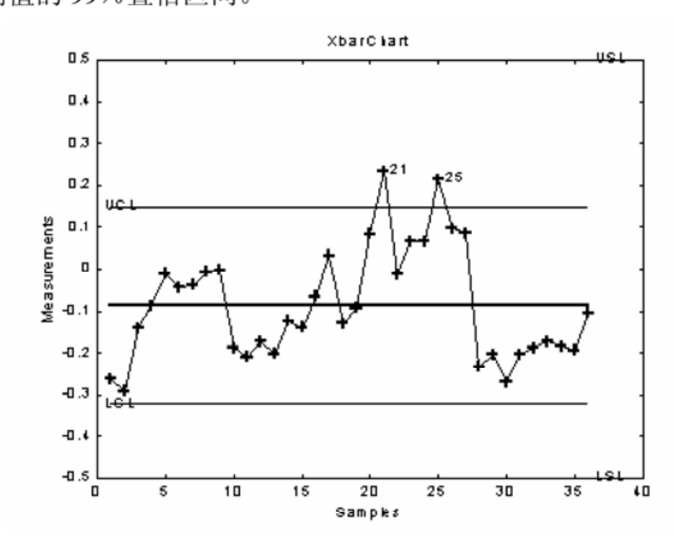


图 1.11.2 生产零件尺寸的均值管理图

xbarplot(DATA,conf)可用 conf 给出自定的置信区间管理限。如果 conf＝0.95 时，图形则绘出 95%置信区间。

xbarplot(DATA,conf,specs)可用两个元素矢量 specs 限制所绘曲线的界限。

[outliers,h] = xbarplot (DATA,conf,specs)给出 DATA 中那些数据均值失控的行的序号(返回至矢量 outliers 中)，还给出曲线的句柄(矢量 h)。

举例：同 schart 中的例子，绘制其均值管理图。

```
load parts
xbarplot(runout,0.999,[-0.5 0.5])
```

参见：capaplot, histfit, ewmaplot, schart。

3 ewmaplot

功能：指数加权滑动平均图(EWMA)。

格式：ewmaplot(data)

ewmaplot(data,lambda)

ewmaplot(data,lambda,alpha)

ewmaplot(data,lambda,alpha,specs)

h = ewmaplot(...)

说明：ewmaplot(data)绘制数据样本的 EWMA 图。data 中的行数据是按时间顺序给出的观测值。

ewmaplot(data,lambda)由参数 lambda 的取值规定当前预测值受此前观测数据影响的程度，从而绘制数据样本的 EWMA 图。lambda 取值区间为[0,1]，缺省值为 0.4。lambda 值越高，过去观测值所占的权重越大。

ewmaplot(data,lambda,alpha)则又规定了所绘制的置信区间的上下界的显著性水平。alpha 的缺省值为 0.01，这意味着图中所绘数据点中大约有 99%落于此控制区间内。

ewmaplot(data,lambda,alpha,specs)中的 specs 矢量(包含两个元素)给出了所绘制的图中对响应结果规定的上下界线。

h = ewmaplot(...)返回各曲线的句柄矢量。

举例：考虑一个均值随时间缓慢漂移的工序。监控这种工序，采用 EWMA 图(参见图 1.11.3)比 $\overline{X}$ -管理图(即均值管理图)更可取。

```
t=(1:30)?
r=normrnd(10+0.02*t(:,ones(4,1)),0.5);
ewmaplot(r,0.4,0.01,[9.3 10.7])
```

参考：Montgomery, Douglas, Introduction to Statistical Quality Control, John Wiley & Sons 1991. p. 299。

参见：xbarplot, schart。

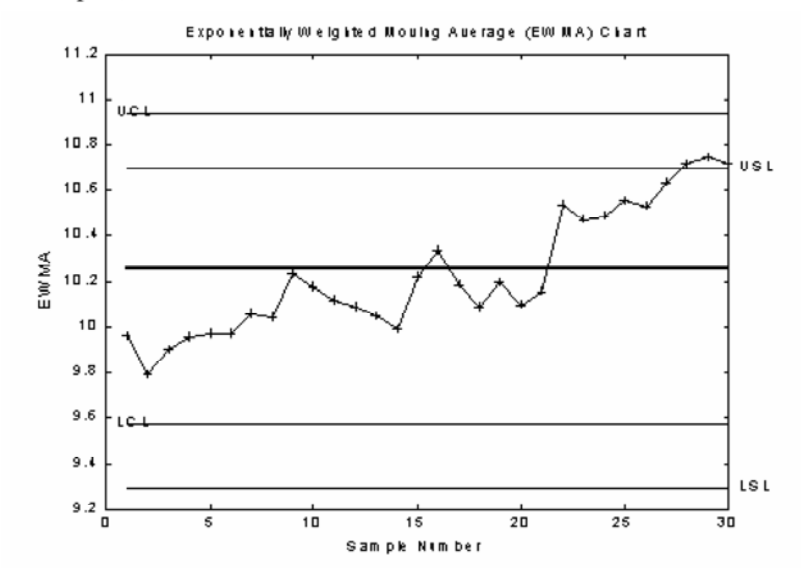


图 1.11.3　工序的指数加权滑动平均图

1.11.3　工序能力

工序能力亦称工序质量，是指工序稳定地生产合格品的能力。工序能力主要表现在稳定性和精度两个方面。工序的稳定性，用管理图来控制；而工序的精度，可用直方图来判断和分析；工序满足质量要求的程度，用工序能力指数表征和分析。

1　hist

功能：绘制直方图。

格式：hist(y)

hist(y,nb)

hist(y,x)

[n,x] = hist(y,...)

说明：hist 函数计算并绘制直方图(参见图 1.11.4)。

hist(y)将矢量 y 中的数据绘制成有 10 条矩形的直方图。

hist(y,nb)绘制具有 nb 条矩形的直方图。

hist(y,x)则按照矢量 x 中给出的矩形分距绘制直方图。

[n,x] = hist(y), [n,x] = hist(y，nb), [n,x] = hist(y，x)并不给出图形，而是给出频率数 n 和矩形的位置。

举例：

```
x = -2.9:0.1:2.9;
y = normrnd(0,1,1000,1);
hist(y,x)
```

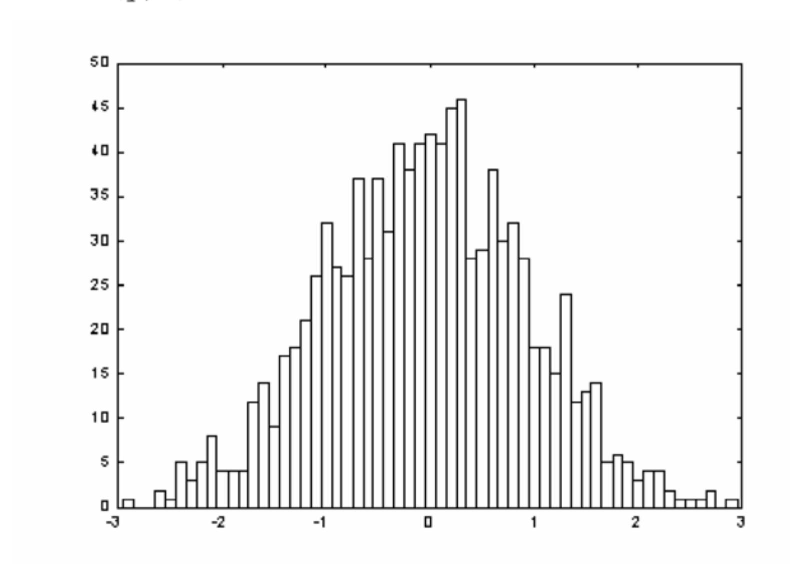


图 1.11.4　正态分布数据的直方图

2 histfit

功能：附加正态密度曲线的直方图。

格式：histfit(data)

histfit(data,nbins)

h = histfit(data,nbins)

说明：histfit(data)将 data 中的数据绘制成直方图(参见图 1.11.5)。此时，图中矩形的条数等于 data 中数据个数的平方根(取大于它的最小整数)。

histfit(data,nbins)在直方图中用 nbins 条矩形绘制 data 的图形。

h = histfit(data,nbins)返回所绘图形的句柄，h(1)为直方图的句柄，h(2)为密度曲线的句柄。

举例：

```
r = normrnd(10,1,100,1);
histfit(r)
```

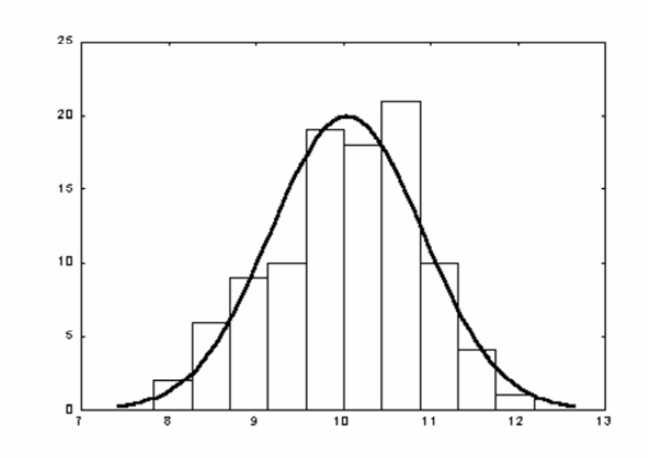


图 1.11.5　附加正态密度曲线的直方图

参见：hist, normfit。

3　normspec

功能：绘制规定区间的正态分布密度曲线。

格式：p=normspec(specs,mu,sigma)

[p,h] = normspec(specs,mu,sigma)

说明: p=normspec(specs,mu,sigma)绘制由矢量 specs 的两个元素所确定的上限和下限间的正态分布密度曲线(参见图 1.11.6)。mu 和 sigma 为此正态分布的参数。

[p,h] = normspec(specs,mu,sigma)除绘制曲线外，还返回样本落于其上限和下限间的概率。h 为曲线对象的句柄。

如果 specs 矢量的第一个元素为－Inf，即下限为－∞；同样，如果 specs 矢量的第二个元素为 Inf，即上限为∞。

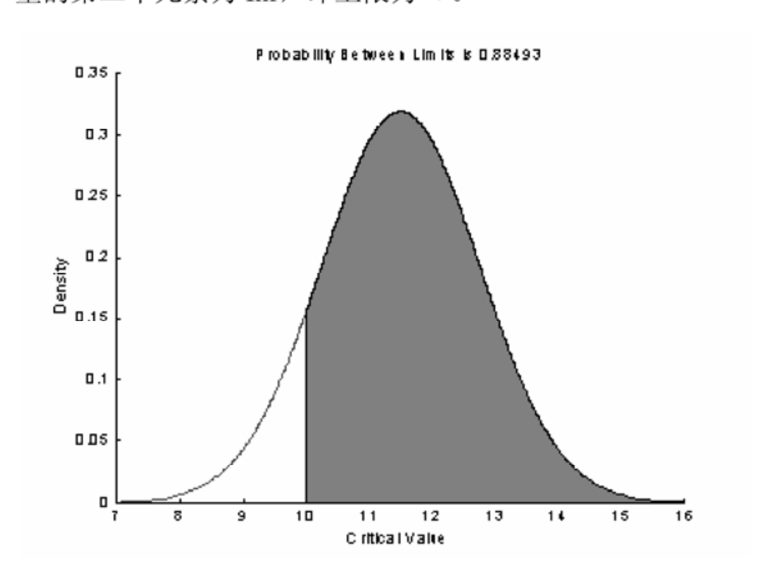


图 1.11.6　规定区间的正态分布密度曲线

举例：谷物加工者生产成箱谷片，每箱重量规定为 10 盎司。但实际装填过程中造成重量偏差，平均每箱谷片的重量为 11.5 盎司,标准差为 1.25 盎司。那么，有多少箱子的重量大于 10 盎司？

```
normspec([10 Inf],11.5,1.25)
```

参见：capaplot, disttool, histfit, normpdf。

4 capable

功能：工序能力指数。

格式：p = capable(data,lower,upper)

[p,Cp,Cpk] = capable(data,lower,upper)

说明：p = capable(data,lower,upper)计算样本数据落于给定上、下界的区间之外的概率。在此，假设数据矢量 data 中的测量值服从正态分布，其均值和方差均为常值，且测量是统计独立的。

[p,Cp,Cpk] = capable(data,lower,upper)则还返回工序能力指数 C_p 和 C_{pk}，它们是工序满足质量要求程度的数值度量。设在工序处于稳定状态下，T 为公差范围 USL−LSL，σ 为质量特征的标准差，μ 为质量特征的分布中心，M 为公差中心。C_p 是给定的公差范围 USL−LSL 与六倍的标准差估计量之比

$$C_p = \frac{USL - LSL}{6\sigma}$$

对于一生产过程，当 μ 大致等于 M 时，C_p =1 相当于每千个产品中有稍多于一个的次品；而当每百万个产品中有一个次品时，其 C_p 值约为 1.6。C_p 值越高，说明工序质量越高。而当 M 与 μ 不等时,则可用 C_{pk} 来描述，其表达式如下

$$C_{pk} = \min\left(\frac{USL - \mu}{3\sigma}, \frac{\mu - LSL}{3\sigma}\right)$$

举例：假设规定一机器零件的尺寸误差在千分之 3 英寸内(误差服从正态分布)。如果在生产过程中，每个零件平均厚了千分之一英寸，同时其标准差为千分之一英寸，则问工序能力指数是多少？

```
data = normrnd(1,1,30,1);
[p,Cp,Cpk] = capable(data,[-3 3]);
indices = [p Cp Cpk]
indices =
0.0172   1.1144   0.7053
```

由结果可知，在每千个零件中，大约有 17 个零件不符合规定要求。

参考：Montgomery, Douglas, Introduction to Statistical Quality Control, John Wiley & Sons 1991. pp. 369-374。

参见：capaplot, histfit。

5 capaplot

功能：工序能力图。

格式：p = capaplot(data,specs)

[p,h] = capaplot(data,specs)

说明：假设数据服从正态分布，其均值和方差未知，p = capaplot(data,specs)将 data 矢量中的观测值拟合成正态分布图(参见图 1.11.7)，并返回新观测值落于 specs 规定的范围内的概率 p。在分布图中，由 specs 矢量中的两个元素所界定的下界和上界之间的图形部分用阴影表示。

[p,h] = capaplot(data,specs)返回所绘图形的元素的句柄。

举例：假设规定一机器零件的尺寸误差在千分之 3 英寸内(误差服从正态分布)。在生产过程中，每个零件平均厚了千分之一英寸，同时其标准差为千分之一英寸。

```
data = normrnd(1,1,30,1);
p = capaplot(data,[-3 3])
p =
0.9784
```

结果表明(参见图 1.11.7)，新观测值落于 specs 规定的范围内的概率 p 为 97.83%。

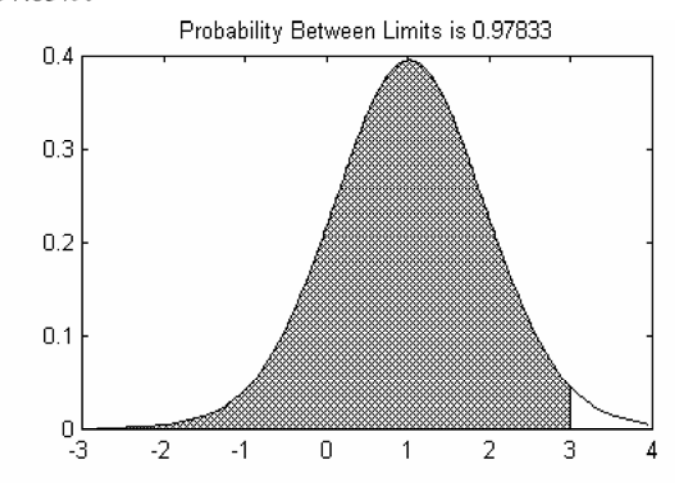


图 1.11.7　机器零件生产的工序能力图

参见：capable, histfit。

1.12　文件输入/输出

统计工具箱提供了几个必要的函数，以数据文件的形式进行数据的输入输出。通过 Windows 标准“打开文件”和“保存文件”对话框，可以方便地操作数据文件。输入输出(I/O)函数见表 1.12.1。

表 1.12.1 输入输出(I/O)函数

函 数 名 称	功 能
caseread	从文件中读实例名
casewrite	将字符串矩阵中的实例名写入文件
tblread	在文件系统中查询表格数据
tblwrite	将表格数据写入文件系统

1 caseread

功能：从文件中读实例名。

格式：names = caseread(filename)

names = caseread

说明：names = caseread(filename)读取 filename 文件的内容并返回实例名称字符串矩阵。其中，filename 是当前目录下一个文件的名称，或者是其他目录下任一文件的完整路径名。caseread 将 filename 文件中每一行看作一个单独的实例名称。

names = caseread 则显示打开文件对话框，可交互选择需要输入的文件。

举例：此例中使用 casewrite 函数参考中创建的文件 month.dat。

```
type months.dat
January
February
March
April
May
names = caseread('months.dat')
names =
January
February
March
April
May
```

参见：tblread, gname, casewrite。

2 casewrite

功能：将串矩阵中的实例名称写入文件。

格式：casewrite(strmat,filename)

casewrite(strmat)

说明：casewrite(strmat,filename)将 strmat 中的内容写入文件。其中，filename 是当前目录下文件的名称，或者是其他目录下文件的完整路径名。strmat 的每一行代表一个实例名称。casewrite 将每一实例名称写入 filename 文件的每一单独的

行。

names = casewrite 则显示打开文件对话框，可交互选择需要输出的文件。

举例：
```
strmat =
str2mat('January','February','March','April','May');
strmat =
   January
   February
   March
   April
   May
casewrite(strmat,'months.dat')
type months.dat
   January
   February
   March
   April
   May
```

参见：gname, caseread, tblwrite。

3 tblread

功能：在文件系统中查询表格数据。

格式：[data,varnames,casenames] = tblread

[data,varnames,casenames] = tblread('filename')

[data,varnames,casenames] = tblread('filename','delimiter')

说明：[data,varnames,casenames] = tblread 显示出打开文件窗口，可交互式选择需要打开的数据表格文件。文件格式为：第一行是变量名，第一列为实例名，数据则始于元素(2，2)。

[data,varnames,casenames] = tblread('filename') 规定了需要打开的文件 filename。Filename 或是当前目录下的文件名，或是其他目录下文件的完整路径。

[data,varnames,casenames] = tblread('filename','delimiter')还可规定表格文件数据域分隔符（'delimiter'）的格式。‘delimiter’可接受的值为‘tab’、‘space’和‘comma’。

- varnames 是包含表格文件第一行（即变量名）的字符串矩阵。
- casenames 是包含表格文件第一列（即实例名）的字符串矩阵。
- data 为对应每一变量-实例对值的数值矩阵。

举例：
```
[data,varnames,casenames] = tblread('sat.dat')
data =
  470 530
```

```
  520 480
varnames =
  Male
  Female
casenames =
  Verbal
  Quantitative
```

参见：caseread, tblwrite。

4 tblwrite

功能：将表格数据写入文件系统。

格式：tblwrite(data,'varnames','casenames')

tblwrite(data,'varnames','casenames','filename')

说明：上述函数语句中的参数的名称含义与 tblread 函数的规定相同，不再赘述。

举例：继续 tblread 中的例子。

```
tblwrite(data,varnames,casenames,'sattest.dat')
type sattest.dat
                 Male          Female
Verbal            470           530
Quantitative      520           480
```

参见：casewrite, tblread。

参 考 文 献

[1] Matlab Statistics Toolbox User's Guide, MathWorks, 1997

[2] 周概容主编，应用统计方法词典，中国统计出版社，1993

[3] 陆璇编著，数理统计基础，清华大学出版社，1998

[4] [美]A.M.穆德，F.A.格雷比尔著，统计学导论，科学出版社，1978

[5] 孙文爽，陈兰祥编，多元统计分析，高等教育出版社，1994

第 2 章　偏微分方程工具箱

(Partial Differential Equqtion Toolbox Ver 1.0)

众所周知，偏微分方程和常微分方程一样，只是在一些特殊情况下才能求得定解问题的精确解。但工程技术和科学实验提出的大量偏微分方程问题需要解决，此时必须借助数值方法或近似方法求得此问题的数值解。Matlab 中的偏微分方程(PDE)工具箱就是用有限法寻求典型偏微分方程的数值近似解。具体地说，该工具箱可以求解线性的椭圆型、双曲型、及抛物线型偏微分方程， 还可求解本征型方程和简单的非线性偏微分方程。本工具箱中的一些函数涉及有限元法的有关概念，因而，先介绍有限元法。

2.1　有 限 元 法

有限元法是近十几年发展起来的一种计算方法，得到广泛应用。它是一种离散方法，将一个连续体剖分为“有限个基本元”，然后用有限个参数描述这个基本元上的物理和力学特性，建立平衡方程，把连续问题转化为离散问题。在有限个点上求出节的近似值。

设在有界区域内讨论方程

$$-\nabla\cdot(c\nabla u)+au=f$$

其中 Ω 是平面上的有界区域，c 、a、f 及 u 是区域 Ω 上的函数，c 也可以是 2*2 的函数矩阵，边界条件包括 Ω 的边界 $\partial\Omega$ 上 u 值及其导数值，一般有如下三种形式。

一是在边界 $\partial\Omega$ 上直接给出未知函数 u 的数值，即

$$u|_T=r \qquad (x, y)\in\partial\Omega$$

这种形式的边界条件称为 Dirichlet 条件或第一类边界条件。

二是在边界 T 上给出未知函数 U 沿 $\partial\Omega$ 的外法线方向导数，即

$$n\cdot(c\nabla u)+qu=g$$

这种形式的边界条件称之为 Neumann 条件或第二类边界条件。

三是对于偏微分方程组，边界条件可以是第一和第二种边界条件的混合，这种形式称为第三类边界条件。

用有限元法求解偏微分方程的过程如下：

(1) 剖分区域，将区域分为有限个互不重叠的三角形“基本元”，根据实际问题的需要在 Ω 内取 N_P 个点，以使这些点连成三角形网络，每个三角形就是一个“基本元”，三角形基本元的顶点称为节点，有一条落在边界的基本元称为边界基本元，其余称为内部基本元；

(2) 将基本元和节点编号，先内部基本元后边界基本元，先内部节点后边界节点；

(3) 在每个基本元(包括边界基本元在内)上构造变分问题解的线性插值函数 ϕ_i (i 为基本元编号);

(4) 将每一个单元上构造的函数 ϕ_i 合并起来，就得到在整个区域 Ω 上的分块近似函数。并且定义

$$K_{i,j}=\int_G (c\nabla\phi_j)\cdot\nabla\phi_i dx$$

$$M_{i,j}=\int_G a\phi_j\phi_i dx$$

$$Q_{i,j}=\int_T q\phi_j\phi_i ds$$

$$F_i=\int_G f_i\phi_i d\mathrm{x}$$

$$G_i=\int_T g\phi_i ds$$

(5) 总体合成

将 $K_{i,j}$、$M_{i,j}$、$Q_{i,j}$、F_i、G_i 合成矩阵 K、M、Q、F、G， 则由变分原理及有限元的理论可知，原微分方程等价于线性方程

$$(K+M+Q)U=F+G$$

其中 K、M、Q 是 N_p*N_p 矩阵，由函数 assema 产生，F、G 是 N_p 维向量，由 assemb 产生，当不必要区分 K 、M、 Q 及 F、G 时，上式也可合并为

$$KU=F$$

偏微分工具箱(PDE ToolBox)利用有限元方法，研究和求解关于 2 维区域和时间 t 的偏微分方程。它的命令函数及其图形界面可以广泛地应用于涉及求解偏微分方程的应用工程科学，包括结构力学、电磁学、热传导及扩散问题。本工具箱具有如下特点：

- 完整的 2 维偏微分方程的预处理和后处理界面；
- 自适应产生有限元网格并可精化网格；
- 利用界面构造相应的几何体；
- 描述边界条件：Dirichlet 条件、广义的 Neumann 条件及混合边界条件；
- 灵活处理偏微分方程的系数及方程描述；
- 可以处理多参数的非线性系统；
- 可以给出方程解的多种特性的可视化结果。

需要指出的是，本工具箱求解偏微分具体步骤与用有限元方法求解偏微分方程的过程是一致的，包括几个步骤，即几何描述、边界条件描述、偏微分方程类型选择、有限元划分计算网格、初始化条件输入，最后给出偏微分方程的数值解(包括画图)。

特别注意的是本工具箱只解决下列类型偏微分方程，即

- 椭圆型偏微分方程

$$-\nabla\cdot(\nabla u)+au=f \quad \text{in } \Omega$$

- 非线性偏微分方程：

$$-\nabla \cdot (c(u)\nabla u) + a(u)u = f(u)$$

• 本征型问题(或特征值问题)

$$-\nabla \cdot (c\nabla u) + au = \lambda du$$

• 双曲型偏微分方程

$$d\frac{\partial^2 u}{\partial t^2} - \nabla \cdot (c\nabla u) + au = f \qquad \text{in} \ \ \Omega$$

• 抛物线型偏微分方程

$$d\frac{\partial u}{\partial t} - \nabla \cdot (c\nabla u) + au = f \qquad \text{in} \ \ \Omega$$

上述方程中的系数 c、a、d、f 可以是关于(x,y)及时间的复杂函数。

可将工具箱中的函数简单地划分为如下 4 类。

(1) 偏微分方程算法函数(见表 2.1.1)

表 2.1.1 偏微分方程算法函数列表

函　　数	功　　能
adaptmesh	生成自适应网格及偏微分方程的解
assemb	生成边界质量和刚度矩阵
assema	生成积分区域上质量和刚度矩阵
assempde	组成偏微分方程的刚度矩阵及右边
hyperbolic	求解双曲线型偏微分方程
parabolic	求解抛物线型偏微分方程
pdeeig	求解特征型偏微分方程
pdenonlin	求解非线性型偏微分方程
poisolv	利用矩形格式快速求解泊松方程

(2) 图形界面函数(见表 2.1.2)

表2.1.2　图形界面函数列表

函　　数	功　能
pdecirc	画圆
pdeellip	画椭圆
Pdemdlcv	转化为版本1.0式的 *.m型文件
pdepoly	画多边形
pderect	画矩形
pdetool	偏微分方程工具箱的图形用户界面

(3) 几何处理函数(见表 2.1.3)

表2.1.3　几何处理函数列表

函　　数	功　　能
csgchk	检查几何矩阵的有效性
csgdel	删除接近边界的小区
decsg	将固定的几何区域分解为最小区域
initmesh	产生最初的三角形网格
Jigglemesh	微调区域内的三角形网格
poimesh	在矩形区域上产生规则的网格
refinemesh	细化三角形网格
wbound	写一个边界描述文件
Wgeom	写一个几何描述文件
pdecont	画轮廓图
pdemesh	画偏微分方程的三角形网格
pdeplot	画偏微分方程的三角形网格
pdesurf	画表面图命令

(4) 通用函数

表2.1.4 通用函数列表

函 数	功 能
pdetriq	三角形单元的品性度量
poiasma	边界点对快速求解泊松方程的“贡献”矩阵
poicalc	规范化的矩形格式的点索引
poiindex	规范化的矩形格式的点索引
sptarn	求解一般的稀疏特征值问题
tri2grid	由三角形格式转化为矩形格式

2.2 区域划分及有限元网格描述函数

用有限元法求解偏微分方程中，关键是网格生成和刚度及右端向量合成问题，PDE 工具箱提供了大量与此有关的函数。

1 pdebound

功能：生成边界 M 文件。

格式：[q , g , h , r] = pdebound (p , e , u , time)

说明：边界 M 文件给定一个 PDE 问题的边界条件。我们能处理的最普遍的边界条件形式为：

$$\underline{h}u = r$$
$$\vec{n}\cdot(\underline{c}\otimes\nabla u)+\underline{q}u = g+\underline{h}'\mu$$

我们以符号 $\vec{n}\cdot(c\otimes\nabla u)$ 表示由元素（i,1）组成的 N*1 矩阵

$$\sum_{j=1}^{N}\left(\cos(\alpha)\,c_{,j,1,1}\frac{\partial}{\partial x}+\cos(\alpha)c_{i,j,1,2}\frac{\partial}{\partial y}+\sin(\alpha)c_{i,j,2,1}\frac{\partial}{\partial x}+\sin(\alpha)c_{i,j,2,2}\frac{\partial}{\partial y}\right)u_j$$

其中，边界向外法线矢量

$$\vec{n} = (\cos(\alpha),\sin(\alpha))$$

对 M 个 Dirichlet 条件 h 为 M*N 矩阵，M ≥ 0。产生的 Neumann 条件包含源项 h', μ。 由此可得 Lagrange 乘数μ，从而可得令人满意的 Dirichlet 条件。需要定义的参数有 q、g、h、r。当 M=0 时，称产生一个 Neumann 边界条件；M=N 时，为一个 Dirichlet 边界条件；而当 0<M<N 时，为一个混合边界条件。[q, g, h, r]=pdebound(p, e, u, time)计算边界 e 的 q、g、h 及 r 的值；矩阵 p、e 是数据网格，e 只需为边界网格的子集。数据网格的详细描述可在 initmesh 输入时获得。输入量 u 和 time 分别用于非线性求解及时间序列算法。如果相应的参数没有输入集合，则 u 和 time 为空矩阵，若 time 是 NaN 并且任何一个函数 q、g、h 和 r 与 time 有关，则 pdepound 必须返回一个大小合理的矩阵，

相应的输出项在所有位置都包含 NaN。解 u 以一个解矢量 u 表示，详细描述可参见 assempde。q 和 g 必须包含每个边界中点值 q、g，这样我们有 `size(q)=[N^2 ne]`，其中 N 为系统维数，ne 是 e 的边界数，`size(g)=[N ne]`。对 Dirichlet 条件，相应的值必须为零。h 和 r 必须包含每个边界的第一个点的值 h、r，接着为每个边界的第二个点的值 h、r，这样我们有 `size(h)=[N^2  2*ne]`，其中 N 为系统的维数，ne 是 e 的边界数，`size(r)=[N2*ne]`。当 M<N 时，h 和 r 必须附加上 N-M 行零。矩阵 q、 h 的元素以列向量的形式保存在 Matlab 矩阵 q、h 中。

举例：对边界条件

u = 2

$$\vec{h}\cdot(c\otimes\nabla u)+\begin{pmatrix}1 & 2\\ 2 & 0\end{pmatrix}u=\begin{pmatrix}3\\ 4\end{pmatrix}+\underline{h}'\mu$$

以下各值保存在 q, g, h 及 r 中

$$q=\begin{bmatrix}1\\ 2\\ \Lambda\\ 2\end{bmatrix}\qquad g=\begin{bmatrix}1\\ 3\\ \Lambda\\ 4\end{bmatrix}$$

$$h=\begin{bmatrix}1 & 1\\ 0 & 0\\ \Lambda & \Lambda\\ -1 & -1\\ 0 & 0\end{bmatrix}\qquad r=\begin{bmatrix}2 & 2\\ \Lambda & \Lambda\\ 0 & 0\end{bmatrix}$$

参见：initmesh, pdegeom, pdesdt, pdeent。

2　pdegeom

功能：建立几何 M 文件。

格式：ne=pdegeom

d=pdegeom(bs)

[x , y]=pdegeom(bs, s)

说明：给定已知边界参数的区域为 2 维的，设区域及边缘符号为唯一正数。边缘各部分不能互相重叠。整个 2 维问题的表述可包含几个不可分割的部分，且各部分有公共边缘。每个区域的边界可由几个边缘片组成。所有边缘各部分的连接处必须与边缘各部分中点相一致。有时我们称边缘部分为边界部分或边沿部分。边界部分位于最小区域联结处的外部边界，边沿部分位于最小区域间的边沿处。

说明几何问题时有如下两种选择。

- 以函数 decsg 建立一个分解几何矩阵，此项工作可由 pdetool 自动完成。用分解几何矩阵将边缘部分限制为直线、圆或椭圆。在工具箱中，几何 M 文件可由分解矩阵替代。
- 建立一个几何 M 文件。通过建立自己的几何 M 文件，可以得到一个与数学函数吻合得很好的几何图形。以下是一个如何建立心形线的例子：`ne=pdegeom`，ne 为边缘分割段数。

d=pdegeom(bs)为只有一列说明每个边缘分割的向量的矩阵，该矩阵被指定在 bs 中。其中：

- 第一行包含参数初始值，
- 第二行包含参数终值，
- 第三行包含左边区域的符号（“左边”是相对于从起始行 1 到结束行 2 设定的方向而言的），
- 第四行包含右边区域的符号。

所有区域联结处的余角赋值为 0。

[x, y]=pdegeom(bs , s)建立边缘分割部分各点的坐标。bs 指定各边缘分割，s 为与之相应的参数值。bs 可以是标量，参数 s 需与曲线长度近似成比例。所有最小区域在边界至少有两个、最好有三个边缘分割。

举例：函数 cardg 定义一个心形线几何图形。

```
r=2(1+cos (ϕ))
function [x, y]=cardg(bs, s)
nbs=4;
if  nargin = = 0
x=nbs;
return
end
dl=[  0    pi/2   pi       3*pi/2
     pi/2  pi    3*pi/2    2*pi;
      1     1      1         1
      0     0      0         0];
if  nargin = = 1
x=dl(:, bs);
return
end
x=zeros(size(s));
y=zeros(size(s));
[m,n]=size(bs);
if  m = = 1    n = = 1 ,
bs=bs*ones(size(s)) ;  expand  bs
```

```
elseif  m = size(s , 1)   n=size(s , 2),
error (   s must be scalar or of same size as s? ) ;
end
nth=400;
th=linspace(0 ,2*pi ,nth);
r=2*(1+cos(th));
xt=r.*cos(th);
yt=r.*sin(th);
th=pdearcl (th ,[xt ;yt] , s , 0 , 2*pi) ;
r=2*(1+cos(th)) ;
x(:)=r. *cos( th ) ;
y(:)=r. *sin( th ) ;
```

用函数 pdearcl 来使参数 s 与弧长成正比，可以通过输入以下语句来验证函数：

```
pdegplot(  ardg?  , axis equal
[p , e , t ]=initmesh(   ardg? );
pdemesh(p , e , t ) , axis equal
```

现在来解决 PDE 问题$-\Delta u = 1$，其几何图形由心形线定义，用 Dirichlet 边界条件：

```
u=assempde (   ardb? , p , e , t , 1 , 0 , 1 ) ;
pdesurf (p , t , u );
```

注意：参数 s 必须与曲线长度近似成正比。所有最小区域在其边界处必须有至少两个、最好三个边缘分割。

参见：initmesh , refinemesh , pdearcl。

3 wbound

功能：写边界条件文件。

格式：fid=wbound(bl,mn)

说明：fid=wbound(bl,mn)写一个边界 M 文件，名字为‘mn.m’。这个边界 M 文件与边界条件矩阵 bl 等价。如果文件不能被写，fid 返回值为-1。bl 描述 PDE 问题的边界条件，它是一个边界条件矩阵，详见条目 assemb。输出文件‘mn.m’是边界 M 文件的名字，详见 pdebound。

参见：decsg,pdegeom,pdebound,wgeom。

4 wgeom

功能：写一个几何描述文件。

格式：fid=wgeom(dl,mn)

说明：fid=wgeom(dl,mn)输出一个几何 M 文件，名字为‘mn.m’。几何 M 文件等价于分解的几何矩阵 dl。如果文件不能被写，则 fid=-1。dl 是分解的几何矩阵，它的格式见 decsg。‘mn.m’是输出的 M 文件的名字，几何 M 文件的格式见

pdegeom。

参见：decsg,pdegeom,wbound。

5 initmesh

功能：产生初始的三角形网格。

格式：[p,e,t]=initmesh(g)

[p,e,t]=initmesh(g,'PropertyName',PropertyValue,...)

说明：[p,e,t]=initmesh(g)利用区域描述函数 g 返回以三角形基本单元的有限元网格，网格尺寸由区域的几何形状决定。参数 g 可以是分解的几何矩阵，也可以是描述区域的几何 M 文件，几何矩阵及几何 M 文件参见 decsg 及 pdegeom。该函数的输出量 p、e、t 是网格数据。在矩阵 p 中的第一行和第二行是网格点 x 坐标和 y 坐标，在边界矩阵 e 中的第一行和第二行包含边界段起始点和终止点的网格点索引，第三行和第四行是相应参数值，第五行是边界段编号，第六行和第七行分别是边界段左边和右边的子区域个数。在三角矩阵 t 中，前三行是三角形单元顶点顺时针顺序的编号索引，第四行是子区域个数。

举例：在 L 型区域生成有限元网格(见图 2.2.1)，则相应网格描述矩阵 p 、e、 t 为

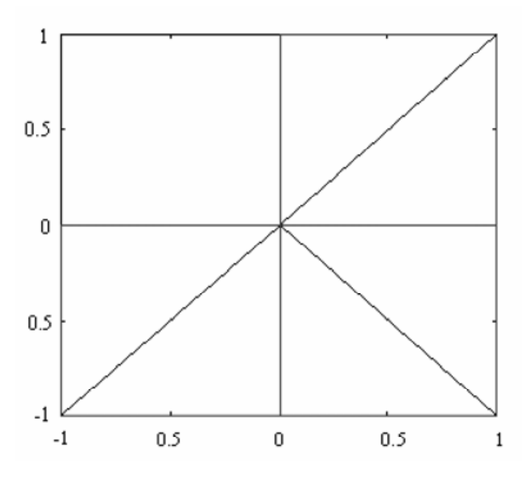


图 2.2.1 有限元网格

```
p =
1 1 0 0
1 1 0 0
e =
1 2 3 4 5 6
2 3 4 5 6 1
0 0 0 0 0 0
1 1 1 1 1 1
1 2 3 4 5 6
1 1 1 1 1 1
```

```
0 0 0 0 0 0
t=1  2  3  1
2  3  4  5
5  5  5  6
1  1  1  1
```

6 refinemesh

功能：精化三角形有限元网格。

格式：[p1,e1,t1]=refinemesh(g,p,e,t)

[p1,e1,t1]=refinemesh(g,p,e,t,'regular')

[p1,e1,t1]=refinemesh(g,p,e,t,'longest')

[p1,e1,t1]=refinemesh(g,p,e,t,it)

[p1,e1,t1]=refinemesh(g,p,e,t,it,'regular')

[p1,e1,t1]=refinemesh(g,p,e,t,it,'longest')

[p1,e1,t1,u1]=refinemesh(g,p,e,t,u)

[p1,e1,t1,u1]=refinemesh(g,p,e,t,u,'regular')

[p1,e1,t1,u1]=refinemesh(g,p,e,t,u,'longest')

[p1,e1,t1,u1]=refinemesh(g,p,e,t,u,it)

[p1,e1,t1,u1]=refinemesh(g,p,e,t,u,it,'regular')

[p1,e1,t1,u1]=refinemesh(g,p,e,t,u,it,'longest')

说明：[p1,e1,t1]=refinemesh(g,p,e,t)返回区域 g 上精化后的网格。g 是区域描述。p 、e 、t 是原网格参数。 函数[p1,e1,t1]=refinemesh(g,p,e,t,u)不仅精化网格，而且利用线性插值方法，求出新网格点上 U1, 参数 it 是指定网格中的需精化子区域，可以是向量或向量组。参数 regula 限制函数精化网格的方式，即由一个三角形基本元生成四个三角形基本元。这种方法也是函数的默认精化方式。Longest 也是一种精化网格方式，它可将指定的三角形最长边划分为两段。

举例：精化L型区域网格。图2.2.2则是L型区域上不同的网格图。

```
[p,e,t]=initmesh('lshapeg','hmax',inf);
subplot(2,2,1), pdemesh(p,e,t)
[p,e,t]=refinemesh('lshapeg',p,e,t);
subplot(2,2,2), pdemesh(p,e,t)
[p,e,t]=refinemesh('lshapeg',p,e,t);
subplot(2,2,3), pdemesh(p,e,t)
[p,e,t]=refinemesh('lshapeg',p,e,t);
subplot(2,2,4), pdemesh(p,e,t)
subplot
```

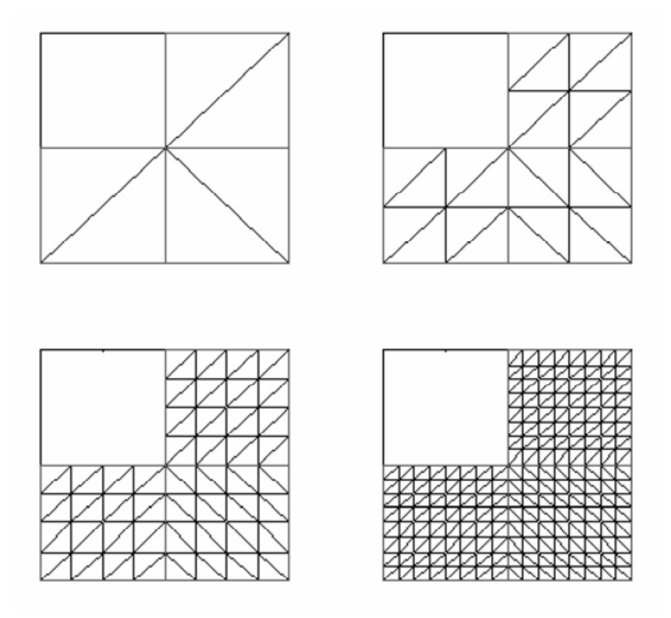


图2.2.2 精化L型区域上不同的网格

7 Adaptmesh

功能：产生自适应的网格和求解 PDE 方程。

格式：[u,p,e,t]=adaptmesh(g,b,c,a,f)

[u,p,e,t]=adaptmesh(g,b,c,a,f, 'propertyname',propertyvalue)

说明：函数[u,p,e,t]=adaptmesh(g,b,c,a,f, 'propertyname',propertyvalue)产生自适应的网格并求解 PDE 方程。可选项由 propertyname/propertyvalue 对给出。

该函数产生一个椭圆标量 PDE 方程

$$-\nabla\cdot(c\nabla u)+au=f on \Omega$$

或椭圆系统方程

$$-\nabla\cdot(c\otimes\nabla u)=f on \Omega$$

同时问题的几何和边界条件由 g 和 b 给出。网格由 p、e 和 t 描述。解以向量 u 的形式给出，关于解向量 miaosh 的详情参见 assempde。解的算法通过用精制三角网格解一系列的 PDE 问题来实现。第一个三角网格的产生通过调用带选项的函数 adaptmesh 或不带选项的函数 initmesh。其后三角网格是通过解 PDE 问题，计算误差，根据误差选择一套三角网格，然后精制它们。最后再计算 PDE 问题的解。循环一直持续到三角网格不再变化或者达到了最大的三角数目，或者产生网格的次数达到了最大。

g 描述 PDE 问题的分解几何。g 可以是一个分解的几何矩阵或几何 M 文件。分解的几何矩阵和几何 M 文件的形式参见 decsg、pdegeom。

b 描述 PDE 问题的边界条件。它是一个边界条件矩阵或边界 M 文件的名字。边界条件矩阵和边界 M 文件的形式参见 assemb 或 pdebound。

网格由网格数据 p、e 和 t 给出，关于网格数据的详情见于条目 initmesh。

PDE 问题的系数 c、a、f 可有很多不同的给法。如果使用给定特性 nonlin，使

用非线性解法，则在 adaptmesh 中系数决定于解 u。系数不依赖于时间 t。完整的选项列表见条目 assempde。

表 2.2.1 给出了 propertyname 和 propertyvalue 对，它们的缺省值和说明。

表 2.2.1　属性及属性值列表

Propertyname	propertyvalue	缺　省　值	说　　明
Maxt	正整数	inf	新网格的最大三角数
Ngen	正整数	10	产生网格的最大次数
Mesh	p1,e1,t1	initmesh	初始网格
Tripick	Matlab 函数	Pdeadworst	三角选择方法
Par	数字	0.5	函数参数
Rmethod	Longest|regular	Longest	三角精制方法
Nonlin	on|off	Off	使用非线性解法
Toln	数字	1e-4	非线性容差
Init	u0	0	非线性初始值
Jac	Fixed|lumped|full	Fixed	非线性 Jacobian 计算
Norm	numeric|inf|energy	Inf	非线性剩余准则

Par 被传递给 tripick 函数，它通常作为解与方程符合程度的容差。

网格只能精制 ngen 次，当网格内的网格数大于 maxt 时，精制也会停止。

p1、e1 和 t1 是输入的网格数据，这个网格作为适应算法最先使用的网格。网格数据表示的方法见 initmesh。如果没有提供初始网格，将会调用没有选项的 initmesh 函数作为初始网格。

三角选择方法 tripick，是一个用户定义的三角选择方法。通过函数 pdejmps 计算误差估计值，三角选择方法将选择在下一次三角网格生成时需要精制的网格。函数被调用时使用参数 p、t、cc、aa、ff、u、errf 和 par。p 和 t 表示当前网格的生成，cc、aa、ff 是当前 PDE 问题的系数，u 是当前解，errf 是计算的误差估计值，par 是函数参数，作为 adaptmesh 的选项。矩阵 cc、aa、ff 和 errf 有 N_t 列，这里 N_t 是当前的三角数目，cc、aa、ff 的行数与输入的项 c、a、f 相同，errf 的行数与系统中的方程数目相同。工具箱中有两个标准的三角选择方法——pdeadworst 和 pdeadgsc。Pdeadworst 选择 errf 超过最坏系数(缺省值为 0.5)的三角，pdeadgsc 以一个相对容差标准选择三角。

精制方法有最长和规则两种，详见 refinemesh。

适应算法也可以解 PDE 方程。对于非线性的 PDE 方程，非线性参数 nonlin 必须设为 on。非线性容差 toln 非线性初始值 u0，非线性 Jocabian 计算结果 Jac 和非线性残余标准 Norm 被传递给非线性解法函数 pdenonlin。关于非线性解法函数的详情见于 pdenonlin。

举例：解一个扇面上的 Laplace 方程，直线上边界条件是 u=0，弧上是 Drichlet 边界条件 u=cos(2/3atan2(y,x))，并且将结果与精确解进行比较。精制网格使用最大误差标准， 直到得到一个具有 500 个三角的网格，见图 2.2.3。

```
[u,p,e,t]=adaptmesh(  irsg?   irsb? 1,0,0,  axt? 500,?
       ripick?   deadworst?   gen? inf);
x=p(1,:);y=p(2,:);
exact=((x.^2+y.^2).(1/3.cos (2/3*atan2(y,x)) );
```

```
max(abs(u-exact))
ans=0.58
size(t,2)
ans= 534
pdemesh(p,e,t)
```

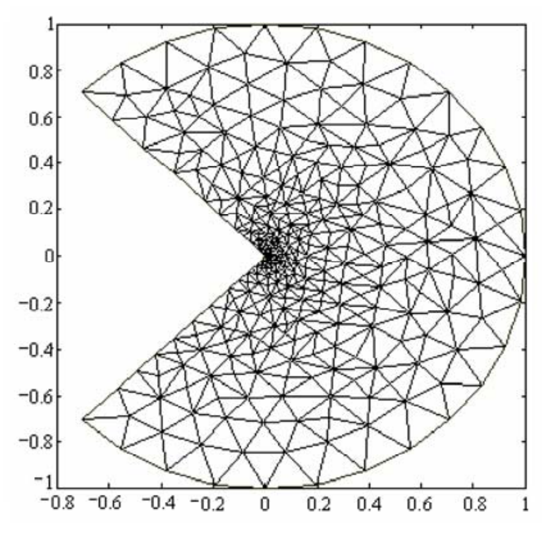


图 2.2.3 精制网格

最大的绝对误差是 0.0058，534 个三角。我们来检验一下做一个规则的三角网格需要多少次精制：

```
[p,e,t]=initmesh(  irsg? ;
[p,e,t,]=refinemesh(  irsg? p,e,t);
u=assempde(  irsb? p,e,t,1,0,0);
x=p(1,:);y=p(2,:);
exact=((x.^2+y.^2).^(1/3).*cos(2/3*atan2(y,x)));
max(abs(u-exact))
ans=0.054
size(t,2)
ans=1640
[p,e,t]=refinemesh(  irsb? p,e,t,1,0,0);
u=assempde(  irsb? p,e,t,1,0,0);
x =p(1,:);y=p(2,:);
exact=((x.^2+y.^2).^(1/3).*cos(2/3*atan2(y,x)));
max(abs(u-exact))
ans=0.054
size(t,2)
ans=6560
pdemesh(p,e,t)
```

这样，用规则的精制网格，我们需要 6560 个三角才能得到我们用适应性网格得到的解的精度。注意规则精制时，当元素的数目增加到四倍，误差只减小 0.6。对一个有规则解的问题，我们希望有 $O(h^2)$的误差，但是这个解是单一的，由于开始 $u \approx r^{1/3}$。

注意：在结束前，将显示下面的一条信息：

- adaption completed(表示 tripick 函数返回零个需要精制的网格)
- Maximum number of triangles obtained
- Maximum number of refinement passes obtained

参见：initmesh,refinemesh,assempde,pdeadgsc,pdeadworst,pdejmps。

8　assema

功能：集合面积分的贡献。

格式：[K,M,F]=assema(p,t,c,a,f)

[K,M,F]=assema(p,t,c,a,f,u0)

[K,M,F]=assema(p,t,c,a,f,u0,time)

[K,M,F]=assema(p,t,c,a,f,u0,time,sdl)

[K,M,F]=assmea(p,t,c,a,f,time)

[K,M,F]=assema(p,t,c,a,f,time,sdl)

说明：[K,M,F]=assema(p,t,c,a,f)集合硬度矩阵 K，质量矩阵 M 和右手向量 F。输入参数 p、t、c、a、f、u0、time 和 sdl 都与 assempde 函数有相同的含义。

参见：assempde。

9　assemb

功能：集合边界条件的贡献。

格式：[Q,G,H,R]=assemb(b,p,e)

[Q,G,H,R]=assemb(b,p,e,u0)

[Q,G,H,R]=assemb(b,p,e,u0,time)

[Q,G,H,R]=assemb(b,p,e,u0,time,sdl)

[Q,G,H,R]=assemb(b,p,e,time)

[Q,G,H,R]=assemb(b,p,e,time,sdl)

说明：[Q,G,H,R]=assemb(b,p,e)集合矩阵 Q 和 H，向量 G 和 R。Q 应当被加入系统矩阵，并且包含混合边界条件的贡献。G 应当被加入右手向量，并且包含 Neumann 和混合边界条件的贡献。方程 H*u=R 代表 Dirichlet 型边界条件。输入参数 p、e、u0、time 和 sdl 与 assempde 函数中的参数有相同的含义。b 描述 PDE 问题的边界条件，它可以是一个边界条件矩阵或者边界条件 M 文件。边界 M 文件的形式见条目 pdebound，边界条件矩阵的形式描述如下。

工具箱可以处理如下类型的边界条件：

- 在一个归一化的 Neumann 边界部分，q 和 g 与下列方程导出的值相关

$$\vec{n}\cdot(c\nabla u)+qu=g$$

- 在一个 Dirichlet 边界部分，hu=r。

工具箱也可以处理在区域Ω上的 PDE 系统。令系统的变量数目为 N，我们通常的边界条件是：

$$\underline{h}u=r$$

$$\vec{n}\cdot(\underline{c}\otimes\nabla u)+\underline{q}u=g+\underline{h}\,\mu$$

这里

$$c\otimes\nabla u$$

是指 N 行 1 列矩阵，矩阵的（i,1）个元素为

$$\sum_{j=1}^{N}(\cos(\alpha)c_{i,j,1,1}\frac{\partial}{\partial x}+\cos(\alpha)c_{i,j,l,2}\frac{\partial}{\partial y}+\sin(\alpha)c_{i,j,2,1}\frac{\partial}{\partial x}+xin(\alpha)c_{i,j,2,2}\frac{\partial}{\partial y})u_j$$

这里α是边界法向量的角度，法向量指向区域Ω的外侧。

边界条件矩阵是由 pdetool 在内部产生的(实际上是 pdetool 的一次调用)，它被函数 assemb 调用，用来集合边界条件矩阵和矩阵 Q、G、H、R 的贡献。边界条件矩阵也可以被存放在一个边界 M 文件中，将来由 wbound 函数调用。对于分解几何矩阵中的每一列，边界条件矩阵中必须有一列与之对应，每一列的组成如下：

第一行是系统的维数；

第二行是 Dirichlet 边界条件的数目 M；

第三行到 $3+N^2-1$ 行是 q 中字符串的长度，顺序是按 q 中列的顺序；

第 $3+N^2$ 行到 $3+N^2+N-1$ 行是 g 的字符串的长度；

第 $3+N^2+N$ 行到 $3+N^2+N+MN-1$ 行是 h 的字符串的长度，顺序也是按 h 的列顺序；

第 $3+N^2+N+NM$ 到 $3+N^2+N+MN+M-1$ 是 r 中字符串的长度。

其余的行包含 Matlab 文本表达式，代表实际的边界条件函数。文本字符串的长度对应上边的描述。Matlab 文本表达式按 h 和 q 中列的顺序存储，字符串中不带间隔符，可在 Matlab 表达式中插入如下变量：

- 二维坐标(x,y)
- 边界元的参数 s，s 与弧的长度成比例，起始处为 0，按箭头所指方向增长，最大值为 1；
- 指向外侧的法向量分量 nx 和 ny，如果需要切向量，也可以用 nx 和 ny 表示，因为

  tx=−ny, ty=nx;
- 解向量 u(此时输入项 u 必须事先区分);
- 时间 t(输入项 time 也需要事先区分)。

举例：以下的例子演示边界条件矩阵的格式。

对于一个标量 PDE 方程(N＝1)的边界，Neumann 边界条件(M＝0)

$$\vec{n}\cdot(c\nabla u)=-x^2$$

边界条件被表示成列向量

[1　0　1　5　'0'　'−x.^2'] '

注意这里没有 h 或 r 的字符串长度。

同样对于一个标量 PDE 方程，Dirichlet 边界条件

$$u=x^2-y^2$$

表示成列向量

[1　1　1　1　9　'0'　'0'　'x.^2−y.^2'] '

对于一个 N＝2，具有混合边界条件

$$(h_{11}\quad h_{12})u=r_1$$

$$\vec{n}\cdot(\underline{c}\otimes\nabla u)+\begin{bmatrix}q_{11} & q_{12}\\ q_{21} & q_{22}\end{bmatrix}u=\begin{bmatrix}g_1\\ g_2\end{bmatrix}+s$$

的系统，其列向量如下：

2

1

1q11

1q21

1q12

1q22

1g1

1g2

1h11

1h12

1r1

q11…

q21…

q12…

q22…

g1…

g2…

h11…

h12…

r1…

这里 1q11、1q21、…表示 Matlab 文本表达式的长度，q11、q21、…是实际的表达式。可以很容易地用 pdetool 生成您自己的例子。在边界模板双击边界选项，然后在‘Boubdary’菜单，选择‘Export Decomposed Geometry,Boundary Cond's…’选项，输出边界条件矩阵到 Matlab 主工作空间。

参见：assempde,pdebound。

10 assempde

功能：生成刚度矩阵和右端向量empde。

格式：
```
u=assempde(b,p,e,t,c,a,f)
u=assempde(b,p,e,t,c,a,f,u0)
u=assempde(b,p,e,t,c,a,f,u0,time)
u=assempde(b,p,e,t,c,a,f,time)
[K,F]=assempde(b,p,e,t,c,a,f)
[K,F]=assempde(b,p,e,t,c,a,f,u0)
[K,F]=assempde(b,p,e,t,c,a,f,u0,time)
[K,F]=assempde(b,p,e,t,c,a,f,u0,time,sdl)
[K,F]=assempde(b,p,e,t,c,a,f,time)
[K,F]=assempde(b,p,e,t,c,a,f,time,sdl)
[K,F,B,ud]=assempde(b,p,e,t,c,a,f)
[K,F,B,ud]=assempde(b,p,e,t,c,a,f,u0)
[K,F,B,ud]=assempde(b,p,e,t,c,a,f,u0,time)
[K,F,B,ud]=assempde(b,p,e,t,c,a,f,time)
[K,M,F,Q,G,H,R]=assempde(b,p,e,t,c,a,f)
[K,M,F,Q,G,H,R]=assempde(b,p,e,t,c,a,f,u0)
[K,M,F,Q,G,H,R]=assempde(b,p,e,t,c,a,f,u0,time)
[K,M,F,Q,G,H,R]=assempde(b,p,e,t,c,a,f,u0,time,sdl)
[K,M,F,Q,G,H,R]=assempde(b,p,e,t,c,a,f,time)
[K,M,F,Q,G,H,R]=assempde(b,p,e,t,c,a,f,time,sdl)
u=assempde(K,M,F,Q,G,H,R)
[K1,F1]=assempde(K,M,F,Q,G,H,R)
[K1,F1,B,ud]=assempde(K,M,F,Q,G,H,R)
```

说明：assempde 是 PDE 工具箱中最基本的函数，它用有限元的方法生成刚度矩阵和右端向量，该函数主要求解下列标量形式的偏微分方程

$$-\nabla\cdot(c\nabla u)+au=f$$

及系统微分方程

$$-\nabla\cdot(c\otimes\nabla u)+au=f$$

标量形式的微分方程解 u 是一个向量，N 维系统微分方程的解是一个向量组，第一列向量是 n_p 个节点上 u_1 分量值，依此类推。

u=assempde(b, p, e, t, c, a, f) 生成偏微分方程的线性系统，并消除其中 Drichilet 条件，进而给出方程解。

[K,F]=assempde(b, p, e, t, c, a, f) 合成偏微分的有限元解法中的刚度矩阵及右端向量 F，方程解为 u=K\F。

[K,M,F,Q,G,H,R]=assempde(b, p, e, t, c, a, f) 将有限元解法的刚度矩阵及右端向量分解为 K、M、F、Q、G、H、R。

u=assempde(K, M, F, Q, G, H, R)则有分解后的 K、M、F、Q、G、H，求出方程解。

参数 b 描述偏微分方程的边界条件，可以是矩阵或 M 文件。关于 b 的格式可参考函数 assemb 或 pdebound。

偏微分方程的几何条件可有 p、e、t 来描述。

标量形式的偏微分方程系数 c、a、f 的表示：

- 常值；
- 由三角形基本元的中心处的取值构成一个向量；
- 或由Matlab函数及程序产生。

对应 N 维偏微分方程组中，c 是一个张量 N*N*2*2，a 是 N*N 的矩阵，f 是长度为 N 的向量，元素 c_{ijkl} ,a_{ij} ,d_{ij} 及 f_i 保存在 Matlab 矩阵 a, c, d, f。当 n_a 为 a 的行数，当 j<i 时，a_{ij}=0；当 j>i 或 i=j 时，其矩阵存放形式见表 2.2.2。

表2.2.2　na取值列表

n_a	对　称　性	元　　素	在矩阵a的行
1	否	a_{ii}	1
N	否	a_{ii}	N
N（N+1）/2	是	a_{ij}	j(j-1)/2+I
N^2	否	a_{ij}	N(j-1)+I

同样，在矩阵 c 中，元素存放格式与系统维数和元素 n_c 有关。表 2.2.3 是其矩阵元素存放格式(其中 j>i 或 j =i，l> k 或 l=k)

表2.2.3　c取值列表

n_c	对 称 性	元素 c_{ijkl}	s所在行
1	否	c_{iikk}	1
2	否	c_{iikk}	K
3	是	c_{iikl}	l+k-1
4	否	c_{iikl}	2l+k-2
N	否	c_{iikk}	I
2N	否	c_{iikk}	2i+k-2
3N	是	c_{iikl}	3i+l+k-4
4N	否	c_{iikl}	4i+2l+k-6
N（2N+1）	否	c_{iikl}	$2j^2$-3j+4i+2l+k-5
$4N^2$	否	c_{ijkl}	4N(j-1)+4i+2l+k-6

举例：求 L 形状区域上的偏微分方程 $-\Delta u = 1$ ，在边界上 u=0,并画出函数解。

```
[p,e,t]=initmesh('lshapeg','Hmax',0.2);
[p,e,t]=refinemesh('lshapeg',p,e,t);
u=assempde('lshapeb',p,e,t,1,0,1);
pdesurf(p,t,u)
```

最后得到的函数解图见图2.2.4。

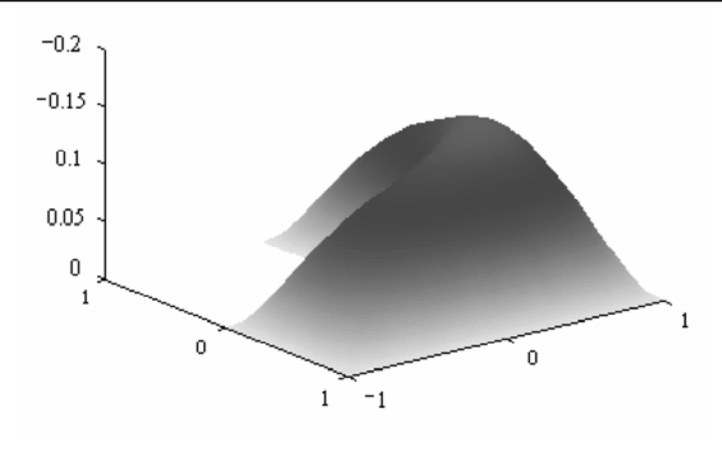


图2.2.4　函数解图

给出偏微分方程组方程个数 N＝3 时， 形式矩阵 a 和 c 在元素个数不同时的存放格式：

n_a=1

a(1)	0	0
0	a(1)	0
0		a(1)

n_a=3

a(1)	0	0
0	a(2)	0
0		a(3)

n_a=6

a(1)	a(2)	a(4)
0	a(3)	a(5)
0		a(6)

同样，当 c 有 n_c 个不同元素时，维数 N＝3 时，其张量矩阵存放格式为：

n_c=1

c(1)	0	0	0	0	0
0	c(1)	0	0	0	0
0	0	c(1)	0	0	0
0	0	0	c(1)	0	0
0	0	0	0	c(1)	0
0	0	0	0	0	c(1)

n_c=3

c(1)	c(3)	0	0	0	0
c(3)	c(2)	0	0	0	0
0	0	c(1)	c(3)	0	0
0	0	c(3)	c(2)	0	0
0	0	0	0	c(1)	c(3)
0	0	0	0	c(3)	c(2)

n_c=12

c(1)	c(3)	0	0	0	0
c(2)	c(4)	0	0	0	0
0	0	c(5)	c(7)	0	0
0	0	c(6)	c(8)	0	0
0	0	0	0	c(9)	c(11)
0	0	0	0	c(10)	c(12)

2.3　求解偏微分方程的函数

PDE 工具箱提供求解抛物线型、双曲线型及本征型偏微分方程的函数。

1　parabolic

功能：求解抛物线型偏微分方程。

格式：u1=parabolic(u0,tlist,b,p,e,t,c,a,f,d)

u1=parabolic(u0,tlist,b,p,e,t,c,a,f,d,rtol)

u1=parabolic(u0,tlist,b,p,e,t,c,a,f,d,rtol,atol)

u1=parabolic(u0,tlist,K,F,B,ud,M)

u1=parabolic(u0,tlist,K,F,B,ud,M,rtol)

u1=parabolic(u0,tlist,K,F,B,ud,M,rtol,atol)

说明：u1=parabolic(u0,tlist,g,b,p,e,t,c,a,f,d)用有限元法求解标量形式的抛物线型偏微分方程

$$d\frac{\partial u}{\partial t}-\nabla\cdot(c\nabla u)+au=f \qquad \text{on } \Omega$$

或偏微分系统方程组

$$d\frac{\partial u}{\partial t}-\nabla\cdot(c\otimes\nabla u)+au=f \qquad \text{on } \Omega$$

网格参数是 p、e、t，边界条件 b 可用矩阵形式也可以用 M 型文件,可依赖于时间 t，方程系数 c 、a、 d、 f 也可以是时间的函数。atol、rtol 是设定的绝对和相对误差。参数 tlist 是一时间序列。对于标量形式的椭圆型偏微分方程，该函数的返回值是一个矩阵，矩阵每一列是对应于网格节点在不同时间时的方程数值解；对于偏微分方程组，则矩阵 u1 中的前 n_p (节点数)对应于 u 的第 1 个分量，依次类推。

u1=parabolic(u0,tlist,K,F,B,ud,M) 用于求解 ODE 问题，即

$$B'MB\frac{du_i}{dt}+Ku_i=F$$
$$u=Bu_i+u_d$$

其中初值为 u0。

举例：求热传导方程

$$\frac{\partial \mathrm{u}}{\partial \mathrm{t}}=\Delta \mathrm{u}$$

求解区域为正方形 $-1\le x,y\le 1$，

初值条件：当 $x^2+y^2\le 1$ 时，u(0)=1，其他情况 u(0)=0。

```
[p,e, t]=initmesh('squareg');
[p,e,t]=refinemesh('squareg',p,e,t);
u0=zeros(size(p,2),1);
ix=find(sqrt(p(1,:).^2+p(2,:).^2)<0.4);
u0(ix)=ones(size(ix));
tlist=linspace(0,0.1,20);
u1=parabolic(u0,tlist,'squareb1',p,e,t,1,0,1,1);
```

经过75步计算；
154次函数赋值；
1次求偏导；
17次LU分解运算；
153组偏微分方程的数值解。

2 hyperbolic

功能：求解双曲线型偏微分方程。

格式：u1=hyperbolic(u0,ut0,tlist,b,p,e,t,c,a,f,d)
u1=hyperbolic(u0,ut0,tlist,b,p,e,t,c,a,f,d,rtol)
u1=hyperbolic(u0,ut0,tlist,b,p,e,t,c,a,f,d,rtol,atol)
u1=hyperbolic(u0,ut0,tlist,K,F,B,ud,M)
u1=hyperbolic(u0,ut0,tlist,K,F,B,ud,M,rtol)
u1=hyperbolic(u0,ut0,tlist,K,F,B,ud,M,rtol,atol)

说明：u1=hyperbolic(u0,ut0,tlist,b,p,e,t,c,a,f,d)用有限元法求解双曲线型偏微分方程

$$d\frac{\partial^2 u}{\partial t^2}-\nabla\cdot(c\nabla u)+au=f \quad \text{on } \Omega$$

或偏微分系统方程组

$$d\frac{\partial^2 u}{\partial t^2}-\nabla\cdot(c\otimes\nabla u)+au=f \quad \text{on } \Omega$$

u0和ut0是初始值和初始导数值。网格参数是p、e、t，边界条件b可用矩阵形式也可以用M文件，可依赖于时间t，方程系数 c、a、d、f 也可以是时间的函数。atol、rtol是设定的绝对和相对误差。

对于标量形式的椭圆型偏微分方程，该函数的返回值是一个矩阵。矩阵每一列是对应于网格节点在不同时间时的方程数值解，对于偏微分方程组，则矩阵u1中的前n_p (节点数)对应于u的第1列分量，依次类推。

u1=hyperbolic(u0,ut0,tlist,K,F,B,ud,M)用于求解 ODE 问题，即

$$B^{/}MB\frac{d^2u_i}{dt^2}+Ku_i=F$$
$$u=Bu_i+u_d$$

其中初值为 u0，初始导数值为 ut0。

举例：求解波动方程

$$\frac{\partial^2 u}{\partial^2}=\Delta u$$

求解区域为-1≦x，y≦1，边界条件：$x=\pm 1$时，u(0)=0，且

$$\frac{\partial u}{\partial n}=0$$

当 $y=\pm 1$ 时，u(0)=arctan(cos(π x))，且

$$\frac{du(0)}{dt}=3\sin(\pi x)\exp(\cos(\pi y))$$

```
c=1;
a=0;
f=0;
d=1;
[p,e,t]=initmesh('squareg');  %产生有限元网格
%初始条件
u(0)=atan(cos(pi/2*x)) and
dudt(0)=3*sin(pi*x).*exp(sin(pi/2*y))
x=p(1,:);
y=p(2,:)';
u0=atan(cos(pi/2*x));
ut0=3*sin(pi*x).*exp(sin(pi/2*y));
%在时间[0,5]上取 31 个点
n=31;
tlist=linspace(0,5,n);
uu=hyperbolic(u0,ut0,tlist,b,p,e,t,c,a,f,d);
%uu 是在下列时刻上的数值解
Time: 0.166667
Time: 0.333333
Time: 0.5
Time: 0.666667
Time: 0.833333
Time: 1
Time: 1.16667
Time: 1.33333
Time: 1.5
Time: 1.66667
Time: 1.83333
Time: 2
Time: 2.16667
```

```
Time: 2.33333
Time: 2.5
Time: 2.66667
Time: 2.83333
Time: 3
Time: 3.16667
Time: 3.33333
Time: 3.5
Time: 3.66667
Time: 3.83333
Time: 4
Time: 4.16667
Time: 4.33333
Time: 4.5
Time: 4.66667
Time: 4.83333
Time: 5
```

经过 428 步运算，982 次赋值。

3 pdeeig

功能：求解特征值问题。

格式：[v, l]=pdeeig(b,p,e,t,c,a,d,r)

[v, l] =pdeeig(K,B,M,r)

说明：[v, l]=pdeeig(b,p,e,t,c,a,d,r) 生成用有限元法求解的特征方程

$$-\nabla\cdot(c\nabla u)+au=\lambda du$$

或特征值系统

$$-\nabla\cdot(c\otimes\nabla u)+au=\lambda du$$

参数p、e、t描述区域，参数b描述边界条件，r是两元素向量，指出在实轴上的一区间(区间左端可为-Inf)，参数l存放该函数返回的区间r上的所有特征值。v是特征向量矩阵，对于标量形式的特征值方程，v对应于网格节点的特征值。对于有n_p个节点的N维系统特征值方程，则v中的前n_p 列(节点数)对于v的第1分量，依次类推，这样v矩阵可分为N块。需注意的是，在偏微分方程的系数g=0、r=0时，与特征值问题是同源问题。

[v, l] =pdeeig(K,B,M,r)主要用于求解希疏矩阵的广义特征值问题，即方程

$$Ku_i=\lambda B^{l}MBu_i \qquad u=Bu_i$$

其中 λ 的实部属于区间r。

举例：求解小方程

$$-\nabla u=\lambda u$$

在L型区域上的小于100的特征值及其相应的特征模态，并演示第十六个特征模态，见图2.3.1。

```
[p,e,t]=initmesh('lshapeg');
[p,e,t]=refinemesh('lshapeg',p,e,t);
[p,e,t]=refinemesh('lshapeg',p,e,t);
[v,l]=pdeeig('lshapeb',p,e,t,1,0,1,[-Inf 100]);
pdesurf(p,t,v(:,16))
```

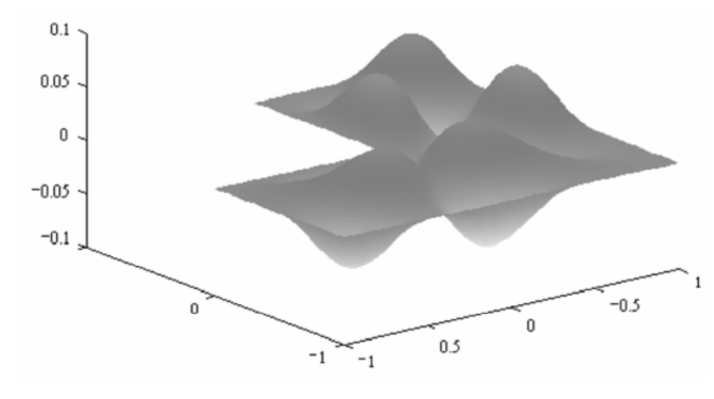


图 2.3.1　特征模态图

注意：1. 当c 和 d 恒为正数时，则所有的特征值都为正。

2. 当d为0时，质量矩阵M为一奇异矩阵，但当c恒大于0时，矩阵束 (K，M) 有限个特征值。

3. 当在一个子区域上等于0时，则K变为奇异矩阵，其有许多0特征值。如果区间上包含0的话，则该函数要花费许多机时寻找0特征值。

4. 当c和d同时为0时，则会得到奇异矩阵束，特征值问题变得没有明确意义，任何值都有可能成为特征值。

4　sptarn

功能：求解一般稀疏特征值问题。

格式：[xv,lmb,iresult]=sptarn(a,b,lb,ub)

[xv,lmb,iresult]=sptarn(a,b,lb,ub,spd)

[xv,lmb,iresult]=sptarn(a,b,lb,ub,spd,tolconv)

[xv,lmb,iresult]=sptarn(a,b,lb,ub,spd,tolconv,jmax)

[xv,lmb,iresult]=sptarn(a,b,lb,ub,spd,tolconv,jmax,maxmul)

说明：[xv,lmb,iresult]=sptarn(a,b,lb,ub,spd,tolconv,jmax,maxmul)找到特征式多项式 $(A-\lambda B)x=0$ 在区间[lb,ub]上的特征值(线性多项式 $A_{ij}-\lambda B_{ij}$,$A-\lambda B$ 组成的矩阵叫特征式串)。A 和 B 是稀疏矩阵，lb 和 ub 分别是要求解的特征值的上界和下界。若要求解 ub 左边的所有特征值，可令 lb=−inf，若要求解 lb 右边的所有特征值，则令 ub=inf。 对于一个窄的区间，可以较快得到结果。复数情况时，比较 lmb、lb、ub 的实数部分。xv 是特征向量，它的值使得判断式

(a*xv−b*xv*diag(lmb))最小。Lmb 是对应的特征值。如果 iresult>=0，则求解是成功的，并且找到了所有的特征值；如果 iresult<0，则求解可能不完全，可能还有更多的特征值(不妨试试更小的区间)。如果已知特征式串的元素均为正值，则 spd=1(缺省值为 0)。tolconv 是期望的相对精度，缺省值是 100*eps，这里 eps 是机器精度。jmax 是基向量的最大数目，求解过程需要 jmax*n 的工作空间，所以在微型计算机上，应该使用较小的 jmax 值，否则可令其等于缺省值 100。正常情况下求解会因得到足够的特征值而提前停止。Maxmul 是 Arnoldi 运行次数，至少应该是所有特征值的倍数。如果 jmax 是一个小值，则较多的 Arnoldi 运行次数是必须的，当单位矩阵的所有特征值被求解时，需使用缺省值，缺省值为 n。

算法：Arnoldi 算法使用光谱变换的方法。变换档选择 ub、lb 或者当两边边界都限定，选择区间(lb,ub)上的一个随机值。

Arnoldi 算法运行的步数 j 依赖于区间上有多少特征值,但是当 j=min(jmax,n)时，计算也会停止。停止后，求解会重新开始，在所有发现的 Schur 向量的补角方向寻找新的向量。

当在区间 lb<lmb<=ub 上不再有新的特征值被发现时，求解停止。

对于较小的 jmax 值，可能需要几次重起，以使计算收敛于某个特征值。当 jmax 的值比区间上的特征值数目大 1 时，求解就可进行，但可能需要很多次重起。对于大值 jmax（推荐使用），需要 mul+1 次运行。Mul 是区间上的特征值的数目的最大倍数。

注意：算法在对称和非对称的 pencil 上都可运行，但在非对称的 pencil 上，其偏离度是正常情况 Henrici 偏离的 tol 倍。参数 psd 只是用来因式分解时在 symmmd 和 colmmd 间选择，对于靠近光谱低端的对称矩阵，前者较好。

可能出现的问题及对策：

- 如果收敛太慢，那么(按下列顺序)

1. 选择一个较小的区间；
2. 选择一个大值的 jmax；
3. 选择一个较大的 maxmul。

- 如果因式分解失败，试着使 lb 或 ub 为有限值，以使得变换为随机，并且不太可能为特征值。如果依然不行，检查是否 pencil 是奇异的。
- 如果运行一直不停止，可能是区间上有太多的特征值，试一下一个小值 maxmul，令其为 2，看看得到的是哪个特征值。一个负的 iresult 说明，那些得到的特征值可能并非全部。
- 如果内存溢出，试试较小的 jmax。
- 算法是为寻找离实轴较近的特征值设计的，如果您想得到离虚轴较近的值，令 A＝i*A。
- 当 spd=1 时，变换档在 lb，因此可以利用对称正定矩阵的快速因式分解。

如果 lb 比最小的特征值大，也没有危害，只是收敛较慢。

参见：pdeeig。

5　pdenonlin

功能：求解非线性偏微分方程。

格式：[u,res]=pdenonlin(b,p,e,t,c,a,f)

[u,res]=pdenonlin(b,p,e,t,c,a,f,'PropertyName', 'PropertyValue',...)

说明：[u,res]=pdenonlin(b,p,e,t,c,a,f)用于求解非线性标量形式的偏微分方程

$$-\nabla\cdot(c\nabla u)+au=f$$

或非线性的偏微分方程组

$$-\nabla\cdot(c\otimes\nabla u)+au=f$$

其中 c 、a 、f 是依赖 u 的函数，该函数主要用牛顿迭代法进行求解。在参数中，主要用于设置方程的迭代次数、迭代终止误差或初解等。需注意的是当该函数不能求解给定方程时，往往会给出迭代次数太多的错误信息。

举例：解最小表面问题，见图 2.3.2。

```
g='circleg';
b='circleb2';
c='1./sqrt(1+ux.^2+uy.^2)';
a=0;
f=0;
rtol=1e-3;
[p,e,t]=initmesh(g);
[p,e,t]=refinemesh(g,p,e,t);
u=pdenonlin(b,p,e,t,c,a,f,'tol',rtol);
pdesurf(p,t,u);
```

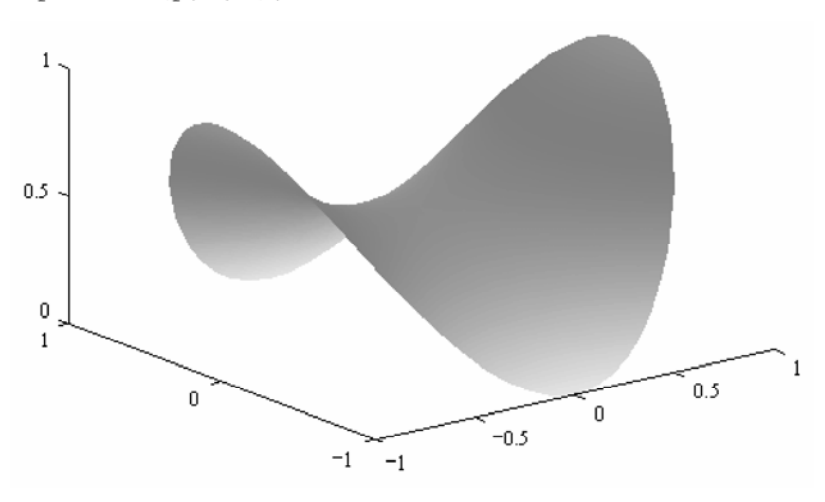


图 2.3.2　最小表面问题

2.4 其他常用函数

1 pdecgrad

功能：求出 PDE 解的通量。

格式：[cgxu, cgyu]=pdecgrad (p , t , c , u)

[cgxu, cgyu]=pdecgrad (p , t , c , u , time)

[cgxu, cgyu]=pdecgrad (p , t , c , u , time , sdl)

说明：[cgxu , cgyu]=pdecgrad (p , t , c , u)返回在每个三角形中心求出的通量

$$\underline{c} \otimes \nabla \underline{u}$$

cgxu 的第 i 行包括

$$\sum_{j=1}^{N} c_{ij11} \frac{\partial u_j}{\partial x} + c_{ij12} \frac{\partial u_j}{\partial y}$$

cgyu 的第 i 行包括

$$\sum_{j=1}^{N} c_{ij21} \frac{\partial u_j}{\partial x} + c_{ij22} \frac{\partial u_j}{\partial y}$$

在 cgxu 和 cgyu 中，对每个三角形而言，t 中都有一列向量。

PDE 问题的几何条件由数据网格 p 和 t 给出。数据网格的具体描述可在 initmesh 输入时获得。

PDE 问题的系数可由不同的方法给出，所有选择的完整列表在 assempde 输入时给出。

解向量 u 的格式在 assempde 中有详细表述。

若 c 依赖于时间 t，标量可选项 time 用于类抛物线和抛物线问题。

可选项 sdl 将计算限定于序列 sdl 的子域中。

参见：assempde。

2 pdecirc

功能：画圆。

格式：pdecirc (xc , yc , radius)

pdecirc(xc , yc , radius , lable)

说明：pdecirc (xc , yc , radius) 画出一个以(xc, yc)为中心，以 radius 为半径的圆。如果 pdtool GUI 不处于被激活的状态，可自动开始，且该圆画在一个空的几何模型中。可选项 label 给该圆命名(否则将选择一个缺省值作为该圆名)。在 pdetool 中几何描述矩阵的状态将被更新，以包含该圆。可用‘Draw’菜单下的命令‘Export Geometry Description …’从 pdetool 输出几何描述矩阵，几何描述矩阵的格式在 decsg 的入口处定义。

举例：以下命令开始 pdetool，并画圆。

```
pdecirc ( 0 , 0 , 1 )
```

参见：pdeellip , pdepoly , pderect , pde tool。

3 pderect

功能：画矩形。

格式：pderect(xy)

pderect(xy,label)

说明：pderect(xy)画一个矩形，顶点坐标由 xy = [xmin xmax ymin ymax]确定。如果 pdetool 的 GUI 没有激活，它会自动打开，矩形画在一个空白的几何模板上。选项 label 将为矩形加上一个名字(否则会有一个默认的名字)。pdetool 中的几何描述矩阵自动更新以包括这个矩形，可以选择‘Draw’菜单下的命令‘Export Geometry Description …’来输出一个几何描述矩阵。在条目‘decsg’中有关于几何描述矩阵的格式的叙述。

举例：以下的一串命令将启动 pdetool，并画出三个正方形连在一起的 L 型平面。

```
pderect([-1 0 ?  0])
pderect([0 1 ?  0])
pderect([0 1 0 1])
```

参见：pdecirc,pdeellip,pdepoly,pdetool。

4 pdepoly

功能：画多边形。

格式：pdepoly(x,y)

pdepoly(x,y,label)

说明：pdepoly(x,y)画一个多边形，顶点坐标由向量 x 和 y 决定。如果 pdetool 的 GUI 没有打开，这时将自动打开，多边形将被画在一个空白的几何模板上。可选项 label 将为多边形命名(否则，将选择一个缺省名)。Pdetool 中的几何描述矩阵将会更新，以包括这个多边形。您可以选择‘Draw’菜单下的命令‘Export Geometry Description …’项来输出一个几何描述矩阵，其格式见 decsg。

举例：

```
pdepoly([-1 0 0 1 1 ? ],[0 0 1 1 ?  ?  ])
```

将生成一个 L 型的平面。

参见：pdecirc,pderect,pdetool。

5 pdecont

功能：快速画等高线的命令。

格式：pdecont (p , t , u)

pdecont (p , t , u , n)

pdecont (p , t , u , v)

h=pdecont (p , t , u ,)

h= pdecont (p , t , u , n)

h= pdecont (p , t , u , v)

说明：pdecont (p , t , u)画 PDE 节点或三角形数据 u 的 10 条水平曲线。

h=pdecont(p , t , u)另外为已有的坐标轴对象返回句柄。如果 u 是一个列向量，则假定为节点数据；若 u 是一个行向量，则假定为三角形单元数据。用函数 pdeprtni 将三角形数据转换为节点数据。PDE 问题的几何条件由网格数据 p 和 t 给出。数据网格的详细表述可在 initmesh 的输入时获得。

pdecont (p , t , u , n)用 n 个平面作图。

pdecont (p , t , u , v)用给定的 v 个平面作图。

如果想更多的控制等高线作图，用 pdeplot 替代 pdecont。

举例：在以 L 型定义的几何层中画方程 $-\Delta u = 1$ 解的等高线，用 Dirichlet 边界条件 $u = 0$(在$\partial\Omega$)，见图 2.4.1。

```
[ p , e , t ] = initmesh (   shapeg? ) ;
[ p , e , t ] = refinemesh (   shapeg? , p , e , t ) ;
u=assempde (   shapeb? , p , e , t , 1 , 0 , 1 ) ;
pdecont ( p , t , u )
```

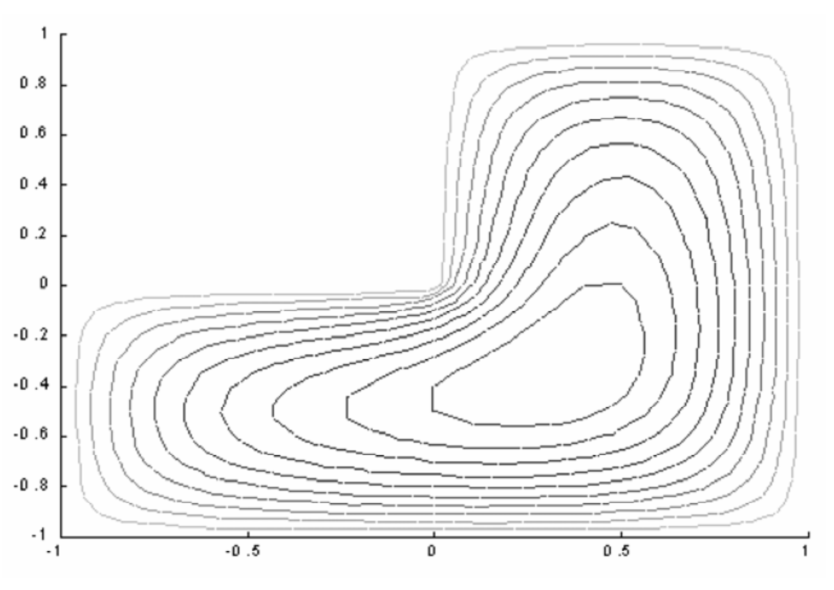


图 2.4.1 解的等高线

参见：pdeplot， pdemesh， pdesurf。

6 pdeellip

功能：画椭圆。

格式：pdeellip (xc , yc , a , b , phi)

pdeellip (xc , yc , a , b , phi , lable)

说明：pdeellip (xc , yc , a , b , phi , lable) 画以(xc , yc)为中心，以 a、b 为半轴的椭圆。椭圆的旋转(弧度)由 phi 给出。如果 pdetool GUI 不处于被激活状态，将

自动开始，且椭圆画于一个空的几何模型中。可选项 label 用于指定椭圆名，否则，将选择一个缺省名。pdetool 中几何描述矩阵的状态被更新，以包含该椭圆。可在 'Draw' 菜单下选择 'Export Geometry Description…' 项从 pdetool 输出几何描述矩阵，几何描述矩阵的格式在 decsg 输入时给出。

举例：以下命令开始 pdetool，且绘出一椭圆。

```
pdeellip ( 0 , 0 , 1 , 0.3 , pi/4 )
```

参见：pdecirc , pdepoly , pderect , pdetool。

7　pdegplot

功能：绘制 PDE 几何图形。

格式：pdegplot (g)

h=pdegplot (g)

说明：pdegplot 绘制一个 PDE 问题的几何图形。

h=pdegplot (g)返回绘制坐标轴对象的句柄。

g 描述 PDE 问题的几何条件。g 是一个分解矩阵，也可以是一个几何 M 文件名。分解矩阵及几何 M 文件的格式分别在 decsg 与 pdegeom 输入时给出。

举例：制 L 型平面的几何图形，见图 2.4.2。

```
pdegplot (   shapeg? )
```

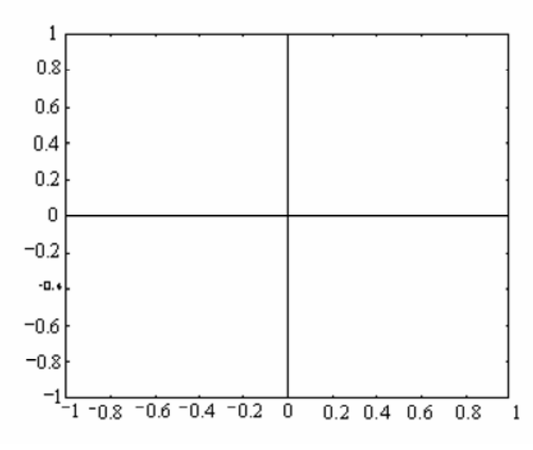


图 2.4.2　几何图形

参见：pdegeom。

8　pdemesh

功能：绘制 PDE 三角网格。

格式：pdemesh (p , e , t)

pdemesh (p , e , t , u)

h=pdemesh (p , e , t)

h= pdemesh (p , e , t , u)

说明：pdemesh (p , e , t)绘制由网格数据 p、e 和 t 定义的网格。

h= pdemesh (p , e , t)另外返回已绘制的坐标对象的句柄。

pdemesh (p , e , t ,u)用一个网格点绘制 PDE 节点或三角数据 u。若 u 是一个列向量，则设为节点数据；若 u 是一个行向量，则设为三角数据。该命令的绘制速度远远大于 pdesurf 命令。PDE 问题的几何条件由网格数据 p、e、t 给出。网格数据的具体描述可见于条目 initmesh。该命令仅用于被调用时快速绘制。

如果想更多地控制您的网格图，用 pdeplot 替代 pdemesh。

举例：绘制 L 型平面的几何网格图，见图 2.4.3。

```
[p , e , t ]=initmesh (   shapeg? );
[ p , e , t ]=refinemesh (   shapeg? , p , e , t ) ;
pdemesh ( p , e , t )
```

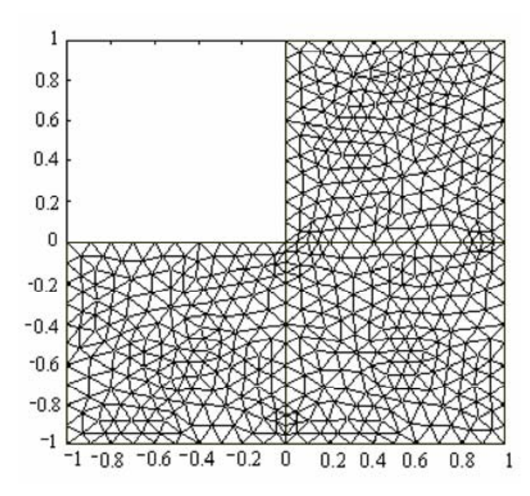


图 2.4.3　L 型平面的几何网格图

现解 Poisson 方程 $-\Delta u = 1$ ，几何条件由 L 形平面定义。用 Dirichlet 边界条件 $u = 0$(在$\partial\Omega$)，且画出结果，见图 2.4.4。

```
u = assempde(   shapeb? , p , e , t , 1 , 0 , 1 );
[p , e , t] = initmesh (   shapeg? ) ;
u=assempde (   shapedb? , p , e , t , 1 , 0 , 1 ) ;
pdeplot (p ,e ,t ,  ydata? u,  data? , u ,  esh? ,  ff? ) ;
```

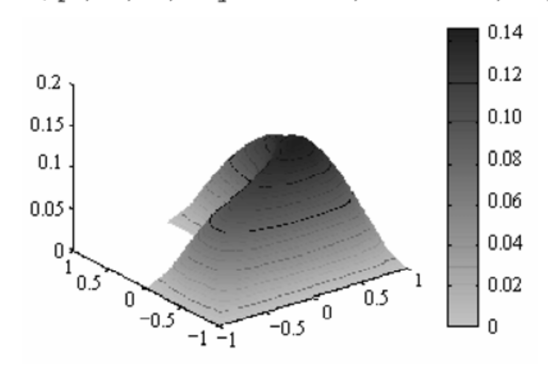


图 2.4.4　Poisson 方程解

9　pdeprtni

功能：由三角形中点值插值求节点的数值。

格式：un=pdeprtni(p,t,ut)

说明：p、t 是三角形网格参数，ut 是数值矩阵。Un 是线性插值后的节点数值。

10　pdesdp, pdesde, pdesdt

功能：子区域的点(或边界 、三角形单元)索引。

格式：
```
c=pdesdp(p,e,t)
[i,c]=pdesdp(p,e,t)
c=pdesdp(p,e,t,sdl)
[i,c]=pdesdp(p,e,t,sdl)
i=pdesdt(t)
i=pdesdt(t,sdl)
i=pdesde(e)
i=pdesde(e,sdl)
```

说明：[i,c]=pdesdp(p,e,t,sdl)中，p、e、 t 是网格参数，sdl 是子区域。

11　Poiasma

功能：求出边界点矩阵对快速求解泊松方程的贡献。

格式：
```
k=poiasma(n1,n2,h1,h2)
k=poiasma(n1,n2)
k=poiasma(n)
```

说明；k=poasma(n1,n2,h1,h2)集合边界点对刚度矩阵的贡献。n1 和 n2 分别是第一和第二个方向的点数，h1 和 h2 是网格间隔。k 是稀疏的 n1*n2 行、n1*n2 列矩阵，点的计数是按照矩形网格的规范计数来进行的。

k =poiasma(n1,n2)时，h1=h2。

k =poiasma(n)时，n1=n2=n。

参见：poiindex,poisolv。

12　pdegrad

功能：求 PDE 方程解的梯度。

格式：
```
[ux,uy]=pdegrad(p,t,u)
[ux,uy]=pdegrad(p,t,u,sdl)
```

说明：[ux,uy]=pdegrad(p,t,u)计算解 u 在每个三角形中心的梯度。ux 的 1 到 N 行为

$$\frac{\partial u_i}{\partial x} \qquad i=1,2,?\ \ N$$

uy 的 1 到 N 行为

$$\frac{\partial u_i}{\partial y} \qquad i=1,2,? \quad N$$

在 ux、uy 中都有一列对应于 t 中的每个三角形。PDE 问题的几何描述通过网格数据 p 和 t 给出，关于网格数据的详情见条目 initmesh。关于解向量 u 的格式的详细描述见 assempde。选项 sdl 限制计算只在 sdl 列出的子域上进行。

参见；assempde。

13　pdesdp, pdesde, pdesdt

功能：给出一个区域上的点/边/三角形的索引。

格式：c=pdesdp(p,e,t)

[i,c]=pdesdp(p,e,t)

c=pdesdp(p,e,t,sdl)

[i ,c]=pdesdp(p,e,t,sdl)

i=pdesdt(t)

i=pdesdt(t,sdl)

i =pdesde(e,sdl)

说明：[i,c]=pdesdp(p,e,t,sdl)如果给定网格数据 p、e、t 和子域数 sdl, 函数将返回属于这些子域的所有的点。一个点可以同时属于几个不同的子域，属于 sdl 中的点被分为不关联的两套。i 包含那些整个属于 sdl 中某些子域的点的索引，c 则包含那些同时属于几个子域的点。C=pdesdp(p,e,t,sdl)返回那些属于 sdl 上多个子域的点的索引。

i=pdesdt(t,sdl)如果给定三角形数据 t 和子域数 sdl，i 将包含所有在这些子域上的三角形的索引。

i=pdesde(e,sdl)如果给定边的数据 e，它将给出属于 sdl 的子域的外边界的边的索引号。如果没有给出 sdl，则默认为全体子域。

14　jigglemesh

功能：微调三角形网格内部点。

格式：pl=jigglemesh(p,e,t)

pl=jigglemesh(p,e,t, 'propertyname',propertyvalue,…)

说明：pl=jigglemesh(p,e,t)通过调整节点位置，微调三角形网格，通常这样做的结果使得网格的质量上升。

表 2.4.1 所列的 propertyname/propertyvalue 对是允许的。

表 2.4.1　属性及属性值列表

名　　称	取　　值	缺 省 值	说　　明
opt	Off | mean | min	mean	优化方法
iter	numeric	1 到 20	最大的迭代次数

每一个不在边缘部分的网格点都移向连起来的三角形形成的多边形的质心。这个过程的重复由变量 opt 和 iter 控制：

- 当 opt 设为 off 时，这个过程重复 iter 次(缺省值为 1)；
- 当 opt 设为 mean 时，这个过程重复至平均的网格质量不再上升为止，或者达到 iter 的上限(缺省值为 20)；
- 当 opt 设为 min 时，这个过程一直进行到最小的三角形的质量不再明显地上升，或者到了 iter 的上限(缺省值为 20)。

举例：在 L 形平面上生成三角形网格，然后再进行微调。

```
[p,e,t]=initmesh( shapeg?  iggle?  ff? ;
q=pdetriq(p,t);
pdeplot(p,e,t, ydata? q, olorbar?  n? ?
  ystyle?  lat?
pl=jigglemesh(p,e,t, pt?  ean?  ter? inf);
q=pdetriq(pl,t);
pdeplot(pl,e,t, ydata? q, olorbar?  n? ?
  ystyle?  lat?
```

参见：initmesh,pdetriq。

15　pdeprtni

功能：在三角形的中点和节点数据间插值。

格式：un=pdeprtni(p,t,ut)

说明：un=pdeprtni(p,t,ut)在三角形的节点和中点间进行线性插值。PDE 问题的几何描述由网格数据 p 和 t 给出，关于网格数据的详细描述见条目 initmesh。设 N 为 PDE 系统的维数，n_p 为节点数目，n_t 为三角形的个数，则 ut 中三角形数据被存为 N 行 n_t 列，un 中的节点数据被存为 N 列 n_p 行。

注意：pdeprtni 和 pdeintrp 不是可逆函数，线性插值导致某种平均。

参见：assempde,initmesh,pdeintrp。

16　pdetriq

功能：度量三角形的品性。

格式：q=pdetriq(p,t)

说明：q=pdetriq(p,t),输入网格数据，则返回一个三角形的质量度量。三角形网格由网格数据 p、e、t 给出，关于它们的详情见条目 initmesh。三角形的品性由下列公式计算

$$q=\frac{4\sqrt{3a}}{h_1^2+h_2^2+h_3^2},$$

这里 a 是面积，h_1、h_2 和 h_3 分别是三角形三条边的边长。

如果 $q>0.6$，则三角形的品性是可取的。当 $h_1=h_2=h_3$ 时，$q=1$。

参见：initmesh,jigglemesh,refinemesh。

17　pdetrg

功能：给出三角形单元的几何数据。

格式：[ar,a1,a2,a3]=pdetrg(p,t)

[ar,g1x,g1y,g2x,g2y,g3x,g3y]=pdetrg(p,t)

说明：[ar,a1,a2,a3]=pdetrg(p,t)可以求出 ar 中的三角形的面积和每个角的半负余切函数值 a2、a2、a3。

[ar,g1x,g1y,g2x,g2y,g3x,g3y]=pdetrg(p,t)返回三角形的面积和三角基函数的梯度分量。PDE 问题的三角形网格由网格数据 p 和 t 给出。

参见：initmesh。

18　tri2grid

功能：在 PDE 问题的三角形网格和矩形网格间插值。

格式：uxy=tri2grid(p,t,u,x,y)

[uxy,tn,a2,a3]=tri2grid(p,t,u,x,y)

uxy=trigrid(p,t,u,tn,a2,a3)

说明：uxy=tri2grid(p,t,u,x,y)将定义在由 p、t 描述的三角网格上的函数 u 转换为定义在由向量 x 和 y 描述的矩形网格上的 uxy。转换按照线性插值进行，向量 x 和 y 必须是增加的。

[uxy,tn,a2,a3]=tri2grid(p,t,u,x,y)除了可以进行上面的运算外，还返回那些包含矩形网格点的三角形的索引号列表 tn 和插值系数 a2、a3。

uxy=tri2grid(p,t,u,tn,a2,a3)先由上面的调用计算出 tn、a2、a3，再在相同的网格上调用 tri2grid 进行插值计算。与一般的调用不同的是，如果有几个函数需要在相同的网格上进行插值，这样做则比较快。如果矩形网格节点出了三角形网格的范围，则 uxy、tn、a2 和 a3 返回值为 NaN。

参见：initmesh,refinemesh,assempde。

19　pdearcl

功能：将弧长描述转换为参数表达。

格式：pp=pdearcl(p,xy,s,s0,s1)

说明：pp=pdearcl(p,xy,s,s0,s1)，输入一个弧长描述的曲线，则返回由参数描述的 pp。p 是包括参数值的单调行向量，xy 是一个两行矩阵，给出曲线上相应的点。曲线的第一点由弧长值 s0 给出，终点由弧长 s1 给出。返回时，pp 包含相应弧长 s 描述的参数值。 弧长 s、s0、s1 可以是弧长的仿射转换。

参见：pdegeom。

20　pdeadworst

功能：根据最坏标准选择三角形。

格式：bt=pdeadworst(p,t,c,a,f,u,errf,wlevel)

说明：bt=pdeadworst(p,t,c,a,f,u,errf,wlevel)返回需要被精制的三角形的号数，被用于adaptmesh 函数选择三角形，以便进一步精制。PDE 问题的几何描述由 p 和 t 给出，详见 initmesh。c、a 和 f 是 PDE 方程的系数，详见 assempde。u 是当前解，以列向量的形式给出，详见 assempde。errf 是误差指示，就像 pdejmps 中计算的一样。wlevel 是相对于最大误差的误差水平，介于 0 和 1 之间。选择三角形的标准是 errf>wlevel*max(errf)。

参见：adaptmesh,pdejmps。

21 poimesh

功能：在矩形面上构造规则网格。

格式：[p,e,t]=poimesh(g,nx,ny)

[p,e,t]=poimesh(g,n)

[p,e,t]=poimesh(g)

说明：[p,e,t]=poimesh(g,nx,ny)在一个由 g 描述的矩形区域上构造规则网格。构造的方法是将“x 边”分成 nx 份,将“y 边”分成 ny 份，在断点处加上(nx+1)*(ny+1)个点。与 x 轴夹角较小的边称为“x 边”。

[p,e,t]=poimesh(g,n)时，取 nx=ny=n；[p,e,t]=poimesh(g)时，取 nx=ny=1。三角形网格由网格数据 p、e 和 t 描述，关于它们的详细描述，请参阅 initmesh。为了较好地使用函数 poisolv，nx 和 ny 中的较大者应是 2 的幂。如果 g 描述的不是一个矩形，p 的返回值为 0。

举例：演示命令 pdedemo8 可以看到一个泊松方程的解，这个方程定义在矩形网格上，边界条件由文件 squareb4 给出。

参见：initmesh,poisolv。

22 pdesmech

功能：计算结构力学的张量函数。

格式：ux=pdesmech(p,t,c,u,'PropertyName',PropertyValue,...)

说明：ux=pdesmech(p,t,c,u,p1,v1,...)返回每一个三角形单元中点处张量表达式，这个张量是应用结构力学中的平面应变及平面应力。输入参数包括偏微分方程的解 u, 网格参数、偏微分方程系数。泊松比参数 nu 主要用于计算剪切应力及剪切应变。

表2.4.2是附加参数名及取值。

表2.4.2　附加参数名及取值列表

<table>
<tr><th>参 数 名</th><th>取值(或默认值)</th><th>说　明</th></tr>
<tr><td>tensor</td><td>{ux|uy|vx|vy|exx|eyy|exy|sxx|syy|sxy|e1|e2|s1|s2|{von Mises}</td><td>张量表达式内容</td></tr>
<tr><td>application</td><td>{ps}|pn</td><td>平面应力|平面应变</td></tr>
<tr><td>nu</td><td>0.3</td><td>泊松比</td></tr>
</table>

张量表达式中各项的意义如下：

- $ux = \dfrac{\partial u}{\partial x}$
- $uy = \dfrac{\partial u}{\partial y}$
- $vx = \dfrac{\partial v}{\partial x}$
- $vy = \dfrac{\partial v}{\partial y}$
- exx是x轴方向的应变
- eyy是y轴方向的应变
- exy是剪应变
- sxx是x方向的应力
- syy是y方向的应力
- sxy是剪应力
- e2是第二主应变
- s1是第一主应力
- s2是第二主应力

在平面应力条件下，von Mises 为

$$\sqrt{\sigma_1^2+\sigma_2^2-\sigma_2\sigma_1}$$

对于平面应变条件，von Mises 则为

$$\sqrt{(\sigma_1^2+\sigma_2^2)(\nu^2-\nu+1)-\sigma_1\sigma_2(2\nu^2-2\nu-1)}$$

23 pdeadgsc

功能：用相对容差的标准选择三角形单元。

格式：bt=pdeadgsc(p , t , c , a , f , u , errf , tol)

说明：bt= pdeadgsc(p , t , c , a , f , u , errf , tol)返回限于 bt 内的三角形编号索引，从而利用自适应精化网格对三角形单元予以细化。PDE 问题的几何条件由网格数据 p , t 给出，详见 initmesh 的输入。c、a、f 是 PDE 的系数，详见 assempde。u 是当前解，以一列向量的形式给出，详见 assempde 的输入。errf 是误差指示，同于 pdejmps 的计算结果。tol 是一个相对容差参数。三角形单元的选择标准是 errf > tol *scale ，其中 scale 由以下计算得出，设 cmax、amax、fmax、umax 分别为 c、a、f、u 的最大值，设 l 为所包含几何区域上的的最小二乘法边界值，则

```
scale=max(fmax * l ^ 2 , amax * umax * l ^ 2 , cmax * umax)
```

scaling 使得 tol 独立于方程和几何条件。

参见：adaptmesh , pdejmps。

24　pdemdlcv

功能：将 PDE 工具箱中 1.0 版本的 M 文件转化为 PDE 工具箱 1.0.2 格式。

格式：pdemdlcv (infile , outfile)

说明：pdemdlcv (infile , outfile)将 PDE 工具箱中 1.0 版本的输入文件转换为 PDE 工具箱 1.0.2 版兼容的 M 文件。转化后 M 文件存为输出文件。如果输出文件中缺‘.m’扩展名，将自动加上。如果想用由 PDE 工具箱 1.0 产生的 M 文件，必须首先用 pdemdlcv 进行转化。

举例：`pdedmdlcv(  odel42.m? ,   odel5.m?`

转换 PDE 工具箱 1.0 格式的文件 model42.m，并将转换结果存于 model5.m 中。

25　pdejmps

功能：估计误差。

格式：errf=pdejmps (p , t , c , a , f , u , alfa , beta , m)

说明：errf=pdejmps (p , t , c , a , f , u , alfa , beta , m)计算误差指数方程。errf 的列向量对应三角形单元，行向量对应 PDE 系统的不同方程。p、t 为网格数据，详见 initmesh 的输入。c、a、f 为 PDE 的系数，详见 assempde 的输入。c、a、f 必须扩展，以使列向量与三角形单元描述方式一致。u 是偏微分方程的解矢量，详见 asempde 的输入。

计算每个三角形 K 的误差指数 $E(K)$的公式为

$$E(K)=\alpha\|h(f-au)\|_K+\beta\left(\frac{1}{2}\sum_{\tau\in\partial K}h_\tau{}^2[n_\tau\cdot(c\nabla u_h)]^2\right)^{1/2}$$

其中 n_τ是边界τ的单位外法向矢量。

参见：adaptmesh , pdeadgsc , pdeadworst。

26　pdetool

功能：调用偏微分工具箱的图形用户界面。

格式：pdetool

pdetool (action, flag)

说明：pdetool 调用偏微分工具箱的图形用户界面(GUI)。不带参数的 pdetool 直接进入应用界面。GUI 可以帮助用户画一个 2 维的平面区域并描述对应于偏微分方程的边界条件，还可以指定偏微分方程类型，生成和精化网格，最后计算出方程的数值解并演示结果。GUI 包括几种不同的模式：

- 在画笔模式中，可以构造出固定的几何体的模型，有四种几何体：
  圆型体——表示圆区域内的点；
  多边形——表示给定边界线段内的多边形区域的点；
  矩形体——表示矩形内的平面区域上的点；

椭圆体——表示椭圆内的区域上的点，并且可将椭圆体旋转。

上述几何体可以移动或旋转，并可剪切或粘贴选定的几何体，还可保存或恢复。另外可以通过键入模板文件名，激活 pdetool。

上述的几何体可以利用公式予以组合。系统自动给每一个几何体分别命名，这些几何体的名字显示在所画的几何体上。组合公式的几何体名字应与所画的几何体相对应。公式中的几何体名表示几何体内的点，组合后的结果仍是相应点的组合。另外该组合公式的默认值是所有几何体的相加。

- 边界模式用来描述微分方程的边界条件。可在不同边界上定义不同的边界条件。在这个模式中，初始几何体的边界可以构成子区域的边界，并且这些边界也可去掉。边界线的颜色代表边界条件类型，红色代表Dirichlet条件，蓝色代表广义Neumann条件，绿色代表混合边界条件。可用按钮‘$\partial\Omega$’显示边界条件，或打开边界条件菜单选择需要的边界条件。
- PDE模式用于指定偏微分方程类型及确定方程系数c、a、d、f，可以指定子区域上的偏微分方程系数。按钮‘PDE’可以显示对话框。
- 在网格模式中，可以控制自动生成网格的方式并画图显示。可用按钮‘Δ’显示初始化的网格，亦可直接用网格菜单的‘Initialize Mesh’项。可用精化网格按钮或直接用精化网格菜单对网格进行细化。
- 在求解模式中，可以指定求解方式并求解偏微分方程。对于抛物线型偏微分方程和双曲型偏微分方程，可以给出初始条件及输出解的时刻点。对于本征型问题，也指定搜索区间。在求解模式中也适合于非线性偏微分方程。
- 在画图模式中，提供了广泛的可视化结果的方法，包括平面图、网格图、等高线图、矢量图等。例如画平面图时，对于抛物线型偏微分方程和双曲型偏微分方程，该模式可以给出方程解随时间变化的动画程。对于一般的方程解，可以很方便地画出平面图或立体图。

边界条件对话框中，可以描述指定边界的边界条件类型，主要处理下列类型的边界条件：

Dirichlet条件： $hu=r$ 在边界上；

广义的Neumann条件： $\vec{n}\cdot(c\nabla u)+qu=g$；

混合边界条件 ： Dirichlet条件和广义的Neumann条件的组合。

偏微分方程描述对话框，主要指定偏微分方程类型和确定方程系数，该对话框适合于下列类型的偏微分方程：

椭圆型偏微分方程

$$-\nabla\cdot(c\nabla u)+au=f \quad \text{in } \Omega$$

非线性偏微分方程

$$-\nabla\cdot(c(u)\nabla u)+a(u)u=f(u)$$

本征型问题(或特征值问题)

$$-\nabla\cdot(c\nabla u)+au=\lambda du$$

双曲型偏微分方程

$$d\frac{\partial u}{\partial t}-\nabla\cdot(c\nabla u)+au=f \qquad \text{in} \ \ \Omega$$

抛物线型偏微分方程

$$d\frac{\partial u}{\partial t}-\nabla\cdot(c\nabla u)+au=f \qquad \text{in} \ \ \Omega$$

参 考 文 献

[1] Partial Differential Equation Toolbox User' Guide, MathWorks, 1997

[2] 陈恕行著，偏微分方程概论，人民教育出版社，1989

[3] 齐民友，徐超江，王维志著，现代偏微分方程概论，武汉大学出版社，1992

第 3 章　样条工具箱

(Spline Toolbox Ver 2.0)

样条(spline)本来是指在飞机和船舶制造过程中为了描绘出光滑的外形曲线所用的一种绘图工具。它是一种富有弹性的细长条，使用时用压铁固定在一些给定的点(称为节点)上，其他地方任它自由变化，然后依样画下的光滑曲线称为样条曲线。从数学上看，它实际上是一段段的三次多项式曲线拼接而成的曲线，在拼接处不仅函数是连续的，而且一阶导数、二阶导数也是连续的，所以样条曲线具有良好的光滑性。

样条函数的概念早在 1946 年就由舍恩伯格(I.J.Schoenberg)提出，至 20 世纪 60 年代研究热潮逐渐兴起，20 世纪 70 年代更是迅速发展起来。它不仅是函数逼近的一个活跃分支，而且也是现代数值计算中一个十分重要的数学工具，已经广泛应用于逼近论、曲线数据拟合、数值积分、数值微分、微分方程和积分方程的数值求解和计算机辅助外形设计与制造等方面。

Matlab 5 中的样条工具箱 2.0 版本为学习样条和使用样条函数提供了一个理想的软件环境。该工具箱提供了样条函数最常用的两种表示形式：B 形式和 PP 形式(Piecewise Polynomial)。B 形式，即以 B 样条函数基底的线性组合来表示样条函数，在构造一个样条函数的过程中非常有用；PP 形式，即以分段多项式来表示样条函数，在评价样条函数时更加有效。工具箱中包含大量对 B 形式和 PP 形式样条函数的操作函数，如创建、显示、插值、逼近和分解等，具体可以分为以下几类：

- 对三次插值样条函数的操作，
- 对 PP 形式样条函数的操作，
- 对 B 形式样条函数的操作，
- 对张量样条函数的操作，
- 其他。

3.1　三次插值样条函数

3.1.1　三次插值样条函数的定义

给定区间$[a, b]$的一个划分

$$\Delta：\ a = x_0 < x_1 < \Lambda < x_n = b$$

和区间$[a,b]$上的一个函数$f(x)$。

定义 **1**　若函数$s(x)$满足下列条件：

1) 一致通过$n+1$个型值点(x_i, y_i)，即

$$s(x_i) = f(x_i) = y_i \quad (i = 0,1,2,\Lambda, n)$$

2) 二阶连续，即

$$s(x) \in C^2[a,b]$$

3) 三次分段，即在每一个小区间$[x_{i-1}, x_i]$上均为三次多项式。

这样的$s(x)$称为$[a,\ b]$上以$x_i(i=1,2,...,n)$为节点的三次插值样条函数，三次插值样条函数的几何图形称为三次样条曲线。

由定义可知，为了确定$s(x)$，可供利用的条件共有$4n-2$,其中插值条件$n+1$个；连续性条件有：

$$s(x_i - 0) = s(x_{\mathrm{i}} + 0)$$

$$s'(x_i - 0) = s'(x_i + 0)$$

$$s''(x_i - 0) = s''(x_i + 0), \text{其中} i = 0,1,2,\Lambda, n$$

共有$3n-3$个。然而$s(x)$是由n段次数不超过 3 的多项式组成，共有$4n$个待定参数。因此为了确定函数$s(x)$，还缺少两个条件。通常是在区间$[a,b]$的端点

$$a = x_0, b = x_n$$

上各附加一个条件，称为边界条件。常见的边界条件有三种：

1) 给定边界点斜率，即

$$s'(x_0) = y'_0, s'(x_n) = y'_n$$

2) 给定边界点的二阶导数，即

$$s''(x_0) = y''_0, s''(x_n) = y''_n$$

3) 给定周期特性，即

$$s^{(\alpha)}(x_0 + 0) = s^{(\alpha)}(x_n - 0)，\ \alpha = 0,1,2$$

3.1.2　三次插值样条函数的构造

记节点处的一阶导数为

$$s'(x_k) = m_k，\quad k = 0,1,\Lambda, n$$

记节点处的二阶导数为

$$s''(x_k)=M_k\,,\qquad k=0,1,\Lambda\ ,n$$

1　用节点处的一阶导数构造三次插值样条函数

由插值条件可得，在$[x_k,x_{k+1}]$上 $s(x)$可表示为

$$s(x)=\left(1-2\frac{x-x_k}{x_k-x_{k+1}}\right)\left(\frac{x-x_{k+1}}{x_k-x_{k+1}}\right)^2 y_k+\left(1-2\frac{x-x_{k+1}}{x_k-x_{k+1}}\right)\left(\frac{x-x_k}{x_{k+1}-x_k}\right)^2 y_{k+1}$$

$$+(x-x_k)\left(\frac{x-x_{k+1}}{x_k-x_{k+1}}\right)^2 m_k+(x-x_{k+1})\left(\frac{x-x_k}{x_{k+1}-x_k}\right)^2 m_{k+1}$$

记 $h_k=x_{k+1}-x_k$，得

$$s(x)=\frac{1}{h_k^3}(x-x_{k+1})^2\left[h_k+2(x-x_k)\right]y_k+\frac{1}{h_k^3}(x-x_k)^2\left[h_k+2(x_{k+1}-x)\right]y_{k+1}$$

$$+\frac{1}{h_k^2}(x-x_{k+1})^2(x-x_k)m_k+\frac{1}{h_k^2}(x-x_k)^2(x-x_{k+1})m_{k+1}$$

于是，只需求出诸节点的斜率即可得到 $s(x)$。令

$$\lambda_k=\frac{h_k}{h_{k-1}+h_k}$$

$$\mu_k=\frac{h_{k-1}}{h_{k-1}+h_k}$$

$$g_k=3\left(\lambda_k f[x_{k-1},x_k]+\mu_k f[x_k,x_{k+1}]\right)\qquad k=1,2,\Lambda\ ,n-1$$

对于给定端点斜率，即 $s'(x_0)=m_0$, $s'(x_n)=m_n$，可以通过解方程组

$$\begin{bmatrix}2 & \mu_1 & & & \\ \lambda_2 & 2 & \mu_2 & & \\ & \mathrm{O} & \mathrm{O} & \mathrm{O} & \\ & & \lambda_{n-2} & 2 & \mu_{n-2}\\ & & & \lambda_{n-1} & 2\end{bmatrix}\begin{bmatrix}m_1\\ m_2\\ \mathrm{M}\\ m_{n-2}\\ m_{n-1}\end{bmatrix}=\begin{bmatrix}g_1-\lambda_1 m_0\\ g_2\\ \mathrm{M}\\ g_{n-2}\\ g_{n-1}-\mu_{n-1}m_n\end{bmatrix}$$

求得诸 m_k。

对于给定端点的二阶导数，即 $s''(x_0)=M_0$, $s''(x_n)=M_n$，可以通过解方程组

$$\begin{bmatrix} 2 & 1 & & & \\ \lambda_1 & 2 & \mu_1 & & \\ & \ddots & \ddots & \ddots & \\ & & \lambda_{n-1} & 2 & \mu_{n-1} \\ & & & 1 & 2 \end{bmatrix} \begin{bmatrix} m_0 \\ m_1 \\ \vdots \\ m_{n-1} \\ m_n \end{bmatrix} = \begin{bmatrix} g_0 \\ g_1 \\ \vdots \\ g_{n-1} \\ g_n \end{bmatrix}$$

求得诸 m_k。式中

$$g_0 = 3f[x_0, x_1] - \frac{h_0}{2} M_0$$

$$g_n = 3f[x_{n-1}, x_n] + \frac{h_{n-1}}{2} M_n$$

对于给定周期特性，可以通过解方程组

$$\begin{bmatrix} 2 & \mu_0 & & & \lambda_0 \\ \lambda_1 & 2 & \mu_1 & & \\ & \ddots & \ddots & \ddots & \\ & & \lambda_{n-2} & 2 & \mu_{n-2} \\ \mu_{n-1} & & & \lambda_{n-1} & 2 \end{bmatrix} \begin{bmatrix} m_0 \\ m_1 \\ \vdots \\ m_{n-1} \\ m_n \end{bmatrix} = \begin{bmatrix} d_0 \\ g_1 \\ \vdots \\ g_{n-2} \\ g_{n-1} \end{bmatrix}$$

求得诸 m_k。式中

$$\mu_0 = \frac{h_{n-1}}{h_{n-1} + h_0}$$

$$\lambda_0 = \frac{h_0}{h_{n-1} + h_0}$$

$$d_0 = 3(\lambda_0 f[x_{n-1}, x_n] + \mu_0 f[x_0, x_1])$$

2　用节点处的二阶导数构造三次插值样条函数

由于 $s(x)$在每个子区间$[x_k, x_{k+1}]$上为三次多项式，所以 $s''(x)$在$[x_k, x_{k+1}]$上为线性函数，记

$$h_k = x_{k+1} - x_k$$

可以推得 $s(x)$在$[x_k, x_{k+1}]$上表示为

$$s(x) = \frac{1}{6h_k}(x - x_k)(x - x_{k+1})(x - 2x_k + x_{k+1})M_{k+1}$$

$$-\frac{1}{6h_k}(x - x_k)(x - x_{k+1})(x + x_k - 2x_{k+1})M_k + \frac{1}{h_k}[(x - x_k)y_{k+1} - (x - x_{k+1})y_k]$$

于是只需求得 $M_k(k=0,1,\ldots,n)$便可以得到 $s(x)$。令

$$\lambda_k = \frac{h_k}{h_{k-1} + h_k}$$

$$\mu_k = \frac{h_{k-1}}{h_{k-1} + h_k}$$

$$d_k = 6f\left[x_{k-1}, x_k, x_{k+1}\right] \qquad k = 1,2,\cdots,n-1$$

对于给定端点斜率，即 $s'(x_0)=m_0$, $s'(x_n)=m_n$，可以通过解方程组

$$\begin{bmatrix} 2 & 1 & & & \\ \mu_1 & 2 & \lambda_1 & & \\ & \ddots & \ddots & \ddots & \\ & & \mu_{n-1} & 2 & \lambda_{n-1} \\ & & & 1 & 2 \end{bmatrix} \begin{bmatrix} M_0 \\ M_1 \\ \vdots \\ M_{n-1} \\ M_n \end{bmatrix} = \begin{bmatrix} d_0 \\ d_1 \\ \vdots \\ d_{n-1} \\ d_n \end{bmatrix}$$

求得诸 M_k。式中

$$d_0 = \frac{6}{h_0}\left(f\left[x_0, x_1\right] - m_0\right)$$

$$d_n = \frac{6}{h_{n-1}}\left(m_n - f\left[x_{n-1}, x_n\right]\right)$$

对于给定端点的二阶导数，即 $s''(x_0)=M_0$, $s''(x_n)=M_n$，可以通过解方程组

$$\begin{bmatrix} 2 & \lambda_1 & & & \\ \mu_2 & 2 & \lambda_2 & & \\ & \ddots & \ddots & \ddots & \\ & & \mu_{n-2} & 2 & \lambda_{n-2} \\ & & & \mu_{n-1} & 2 \end{bmatrix} \begin{bmatrix} M_1 \\ M_2 \\ \vdots \\ M_{n-2} \\ M_{n-1} \end{bmatrix} = \begin{bmatrix} d_0 - \mu_1 M_0 \\ d_2 \\ \vdots \\ d_{n-2} \\ d_{n-1} - \lambda_{n-1} M_n \end{bmatrix}$$

求得诸 M_k。

对于给定周期特性，可以通过解方程组

$$\begin{bmatrix} 2 & \lambda_0 & & & \mu_0 \\ \mu_1 & 2 & \lambda_1 & & \\ & \ddots & \ddots & \ddots & \\ & & \mu_{n-2} & 2 & \lambda_{n-2} \\ \lambda_{n-1} & & & \mu_{n-1} & 2 \end{bmatrix} \begin{bmatrix} M_0 \\ M_1 \\ \vdots \\ M_{n-2} \\ M_{n-1} \end{bmatrix} = \begin{bmatrix} d_0 \\ d_2 \\ \vdots \\ d_{n-2} \\ d_{n-1} \end{bmatrix}$$

求得诸 M_k。式中

$$\mu_0 = \frac{h_{n-1}}{h_{n-1} + h_0}$$

$$\lambda_0 = \frac{h_0}{h_{n-1} + h_0}$$

$$d_0 = \frac{6}{h_0 + h_{n-1}}\left(f\left[x_0, x_1\right] - f\left[x_{n-1}, x_n\right]\right)$$

3.1.3　工具箱中关于三次插值样条的函数

样条工具箱提供表 3.1.1 的函数来构造不同边界条件和要求的三次插值样条函数。

表 3.1.1 关于三次插值样条的函数

函　数　名	函 数 功 能
Csape	构造各种边界条件下的三次插值样条函数
csapi	构造　ot-a-knot'边界条件下的三次插值样条函数
csaps	构造不同光滑程度三次插值样条函数
cscvn	构造普通的或周期的三次样条曲线
getcurve	交互式的创造三次样条曲线

1　csape

功能：构造各种边界条件下的三次插值样条函数。

格式：pp=csape(x, y, [, conds[, valconds]])

说明：(x, y)是插值点的序列，pp 为指定 conds 条件下以(x, y)为插值点所返回的 pp 形式的三次样条函数。

conds 是字符串类型，为边界条件；可为下列字符串或字符串的第一个字母：‘complete’，‘not-a-knot’，‘periodic’，‘variational’，‘second’。它们的含义如下：

‘complete’——给定端点的斜率，斜率大小在 valconds 参数中给出，若忽略 valconds 参数，则按缺省情况处理；

‘not-a-knot’——两个端点存在三阶连续导数(如果给定 valconds 参数则忽略)；

‘periodic’——给定周期特性；

‘second’——给定端点的二阶导数，大小在 valconds 参数中给出，若忽略 valconds 参数，则使用缺省值[0，0]；

‘variational’——给定端点的二阶导数，且大小皆为 0(如果给定 valconds 参数则忽略)；

缺省情况——使用各个端点附近的四个数据计算斜率。

当给定 conds 为一个1×2的矩阵时，可使两个边界点使用不同的边界条件。conds(i)=j 含义是给定端点 i 的 j 阶导数，i=1 是指左端点，i =2 是指右端点，j 一般取 0，1，2，若不是，则取缺省值 1；并且使用 valconds (i)的值指定 conds(i)所指定的 j 阶导数的大小，若忽略该参数则取缺省值。下面介绍 conds 和 valconds 所能取的缺省值，以及取定 conds 时 valconds 所能取的缺省值，见表 3.1.2。

表 3.1.2　缺省值表

lagrange	$Ds(e)=Dp(e)$	缺省
clamped	$Ds(e)=valconds(\ )$	Conds() = 1
variational	$D^2s(e)=0$	Conds()=2 和 valconds(j)=0
periodic	$D^r s(a)=D^r p(b)$,r=1,2	Conds()=[0 0]
curved	$D^2s(e)=valconds(\ )$	Conds() = 2

表中 j = 1 (j = 2)时 e 为第一个(最后一个)节点 a(b)，p(在 Lagrange 边界条件下)是点 e 及其附近三个点的插值多项式。同时该函数还可应用于多变元的样条函数，这里不再详细叙述，可以参考后面的举例。

举例：(1)　单变元

`csape(x,y)` 构造了Lagrange边界条件下的三次插值样条函数；

`csape([-1 1],[-1 1],[1 2],[3 6])`构造了三次多项式p，并使之满足条件

$$p(-1)=-1 \qquad Dp(-1)=3$$

$$p(1)=1 \qquad D^2p(-1)=6$$

(2)　多变元

```
x = 0:4;
y=-2:2;
s2 = 1/sqrt(2);
clear v;
v(3,:,:) = [0 1 s2 0 -s2 -1 0].'*[1 1 1 1 1];
v(2,:,:) = [1 0 s2 1 s2 0 -1].'*[0 1 0 -1 0];
v(1,:,:) = [1 0 s2 1 s2 0 -1].'*[1 0 -1 0 1];
sph = csape({x,y},v,{'clamped','periodic'});
values = fnval(sph,{0:.1:4,-2:.1:2});
surf(squeeze(values(1,:,:));
```

```
squeeze(values(2,:,:));
squeeze(values(3,:,:));
axis equal;
axis off;
```

其中的surf和fnval命令可以简化为fnplt(sph)，图形见图3.1.1。

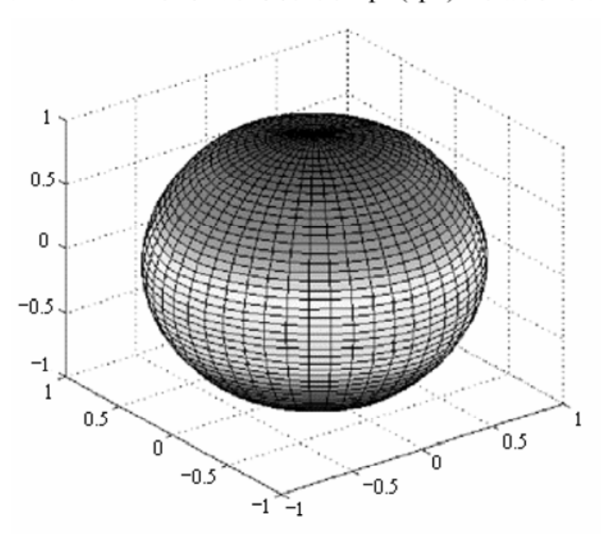


图3.1.1　样条sph的图形

2　csapi

功能：构造‘not-a-knot’边界条件下的三次样条函数。

格式：values=csapi(x, y, xx)

pp=csapi(x, y)

说明：第一种格式下返回的是以(x, y)为插值点序列，以‘not-a-knot’为边界条件的三次插值样条函数在向量 xx 处的值，并存放在 values 中；第二种格式下返回的是以(x, y)为插值点序列，以‘not-a-knot’为边界条件的 PP 形式的三次插值样条函数。该函数还可应用于多变元的样条函数，这里不再详细叙述，可以参考后面的举例。

举例：(1)　单变元

```
x=0:0.2:pi;
y=sin(x);
xx=0:0.5:pi;
pp=csapi(x,y);
values=csapi(x,y,xx);
```

(2)　多变元

```
x =.0001+[-4:.2:4];
y = -3:.2:3;
[yy,xx] = meshgrid(y,x);
r = pi*sqrt(xx.^2+yy.^2);
```

```
z = sin(r)./r;
bcs = csapi( {x,y}, z );
fnplt( bcs );
axis([-5 5 -5 5 -.5 1]);
```

其图形为 3.1.2。

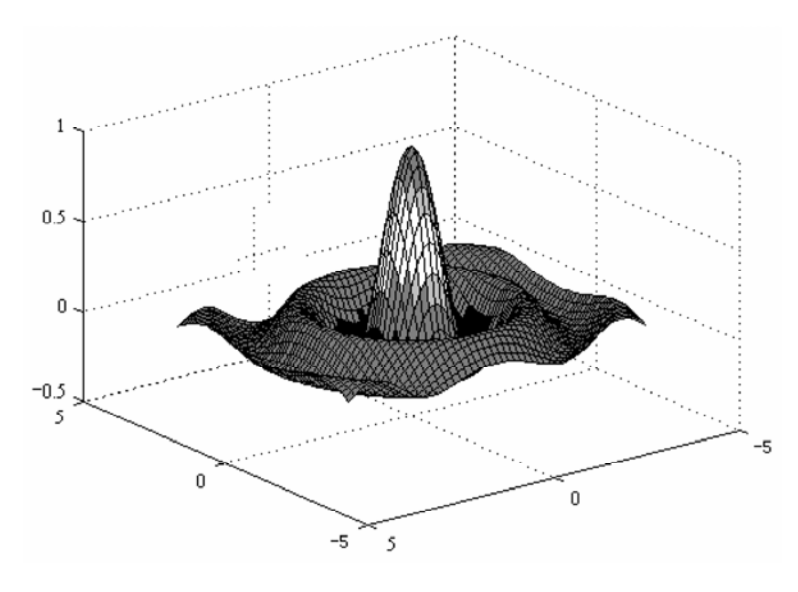


图3.1.2 csapi构造的多变元样条的图形

3 csaps

功能：构造光滑的三次插值样条函数。

格式：values=csaps(x, y, p, xx)

pp=csaps(x, y, p)

说明：第一种格式下返回的是以(x, y)为插值点序列，以 p 为光滑参数（p∈[0, 1]）的三次插值样条函数在 xx 处的值。当 p=0 时以最小方差直线来拟合数据，构造三次插值样条函数，这时光滑程度最低；当 p=1 时以‘variational’为边界构造三次插值样条函数，这时光滑程度最高。第二种格式与第一种格式大致相同，不过返回的是 PP 形式的三次插值样条函数。

举例：

```
x=0:0.2:pi/2;
y=sin(x);
xx=0:0.5:pi;
pp1=csaps(x,y,1);
```

图见 3.1.3。

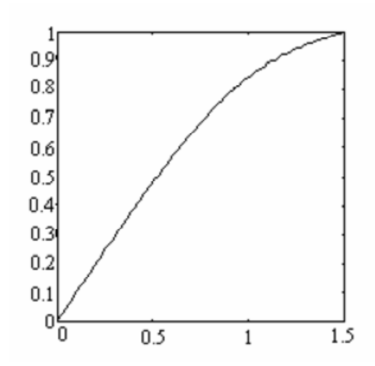


图 3.1.3　csaps构造的的三次插值样条函数

4　cscvn

功能：构造普通的或周期的三次样条曲线。

格式：　cs=cscvn(points)

说明：当 points [:, i]= points [:, end]时，该函数以如下格式调用 csape

cs = csape (t，points，'peiodic')

当 points [:, i]≠points [:, end]时，该函数以如下格式调用 csape

cs=csape (t, points, 'variational')

上两式中 t = cumsum([0, dt.^(1/4)])。

举例：(1)　一般情况

```
points=[0 1 1 0 -1 -1 0 0 0 0 1 2 1 0 -1 -2];
fnplt(cscvn(points));
hold on;
plot(points(1,:),points(2,:),'o');
hold off;
```

图形见图3.1.4。

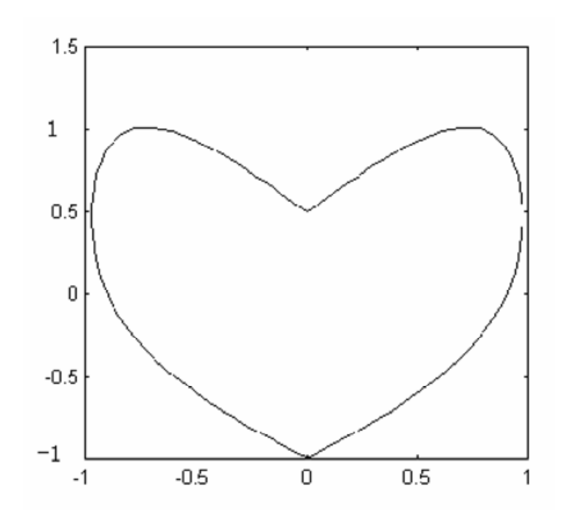


图3.1.4　一般情况

(2) 周期情况

```
fnplt(cscvn([0 .8 .9 0 0 -.9 -.8 0; .5 1 0 -1 -1 0 1 .5])),
```

图形见图3.1.5。

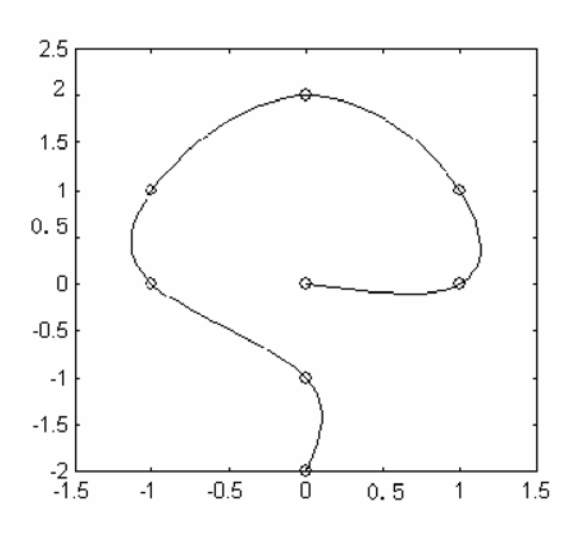


图 3.1.5 周期情况

5 getcurve

功能：交互式的创造三次样条曲线。

格式：[xy, spcv] = getcurve

说明：getcurve显示一个网格划分的窗口，并且要求用户输入。当用户输入点时，函数用折线连接这些点；当用户输入完毕，点击窗口外即可，然后函数返回节点序列xy和用cscvn函数构造的样条函数及其曲线。如果节点序列xy的第一点和最后一点足够接近，那么函数将返回闭合的样条函数和曲线。

3.2 PP 形式的样条函数的构造及操作

通过平面上 $n+1$ 个不同的已知点(x_i,y_i)能作一条光滑曲线，但是在点很多时，插值多项式的次数增高，计算变复杂，逼近程度也不理想。因此，人们采用分段低次插值逼近的办法去克服这一缺点。若用分段线性插值函数，也就是折线代替原曲线，或用分段抛物线去逼近被插函数，也就是利用分段二次曲线去代替原函数，把这种插值函数的方法叫分段插值，所得函数为分段多项式形式的样条函数。

3.2.1 分段多项式形式的样条函数

定义 2 给定区间$[a,b]$的一个划分

$$\Delta：\ a = x_0 < x_1 < \Lambda\ < x_n = b$$

当 $s_n(x)$满足下列两个条件：

1)　在每个子区间$[x_{i-1}, x_i]$ (i=1,2,...,n)上，函数 $s_n(x)$是次数不高于 n 的多项式，但至少有一个子区间上为 n 次多项式；

2)　$s_n(x)$在区间$[a, b]$上存在 n □1 阶的连续导数，即

$$s_n(x) \in C^{n-1}[a,b]$$

则称 $s_n(x)$为关于划分 Δ 的 n 次样条函数，称点 x_i (i=1,2,...,n)为样条函数 $s_n(x)$的节点，x_i (i=1,2,...,n)为样条函数 $s_n(x)$的内节点。

特别当 $a=-\infty$或 $b=+\infty$的无穷区间上，样条函数的上述定义也是适用的。

由定义可知，多项式样条是具有整体某些连续性质的分段多项式。当 n=0 时，条件(2)不起作用，零次样条是阶梯函数，一次样条是折线函数;特别在 3.1 节中所描述三次插值样条函数是取 n=3 时的情况，由于其应用广泛，所以受到特别重视。一般来说，$s_n(x)$在相邻子区间中的多项式不相重，但定义中并未规定它们必须相异。因此 n 次样条函数包含了同次多项式作为它的特例，或是说多项式样条是多项式的一种自然推广。

定义 **3**　给定区间$[a,b]$的一个划分

$$\Delta：\ a = x_0 < x_1 < \Lambda\ < x_n = b$$

当 $s_{n,v}(x)$满足下列两个条件：

1)　在每个子区间$[x_{i-1}, x_i]$ (i=1,2,...,n)上，函数 $s_{n,v}(x)$是次数不高于 n 的多项式，但至少有一个子区间上为 n 次多项式；

2)　$s_{n,v}(x)$在区间$[a,b]$上存在 $n-v$ 阶的连续导数，即

$$s_{n,v}(x) \in C^{n-v}[a,b]$$

则称 $s_{n,v}(x)$为关于划分 Δ 的亏数为 v 的 n 次样条函数。

由定义 2 可知，下述函数

$$s_n(x) = s_{n,1}(x)$$

是亏数为 1 的 n 次样条函数。

3.2.2　工具箱中关于 PP 形式样条函数的函数

在 Matlab 提供的样条工具箱中，分段多项式形式的样条函数 f，即 PP 形式的样条函数，是由它的节点序列 breaks 和局部的多项式片段的参数阵列 coefs 来表示的。节点序列 breaks 被认为是严格增长的，即

breaks(1) < breaks(2) < ⋯< breaks(l+1)

其中 l 是组成样条函数 f 的分段多项式的段数，并且这些多项式的次数可以是不同的，但是它们具有同样的阶数 k，例如参数阵列 coefs 的大小为[l，k]，且 coefs(j,・)为按降幂方

式排列包含第 j 段多项式系数的向量。

PP 形式的样条函数 f 就是在基本区间[breaks(1)，breaks(l+1)]上由 breaks、coefs、l 和 k 四个部分构成的。从这种意义上说，PP 形式的样条函数 f 可按以下方式精确地表示为：

```
f(t) = polyval(coefs(j,:), t < breaks(j));
```

其中，breaks(j) < t < breaks(j+1)。

这里 polyval(a, x)是 Matlab 的自带函数，它的返回值为

$$\sum_{j=1}^{k} a(j)x^{k-j} = a(1)x^{k-1} + a(2)x^{k-2} + \Lambda + a(k)x^{0}$$

当 t 不在区间[breaks(1), breaks(l+1)]内时，f(t)应被扩展定义。例如

```
f(t) = polyval(coefs(1,:),t<breaks(1))
```

其中t < breaks(1)。

PP 形式的样条函数可以通过一些插值、逼近和转换过程来构造，输出为一个行向量，但是也可以用以下方式来拼装：

```
pp=ppmak(breaks,coefs)
```

例如令

```
breaks = -5: -1,
coefs =-22: -11,
```

得到

```
pp=ppmak(-5: -1, -22: -11)
```

其中节点序列 breaks 包含五个元素，将区间分为四段，参数阵列 coefs 包含十二个元素，由此可以确认阶数

```
k = 3 ( = 12/4)
```

用命令

```
fnbrk(pp)
```

可以得到如下结果：

```
breaks(1:l+1)
    -5 -4 -3 -2 ?
coefficients(d*l,k)
    -22 -21 -20
    -19 -18 -17
    -16 -15 -14
    -13 -12 -11
pieces number l
    4
dimension d of target
    1
```

以下是另外一些可以操作 pp 形式样条函数的命令：

```
v = fnval(pp,x)
ipp=fnint(pp),
pj=fnbrk(pp,j),
pc=fnbrk(pp,[a b]),
fnplt(pp),
sp = fn2fm(pp,  -? ,
pr = fnrfn(pp,morebreaks)
```

这些函数、命令将在以后的部分陆续介绍。图 3.2.1 是刚才所构造的样条函数的图形。其绘制过程如下：

```
x = [-55:-5]/10;
plot(x, fnval (pp,x),'-.');
breaks=fnbrk(pp,'b');
yy=axi;
for j=1:fnbrk(pp,  ? +1
   plot(breaks([j j]),yy(3:4));
end
plot(x,fnval(fnbrk(pp,3),x));
set(gca,'ylim',[-60 -10])。
```

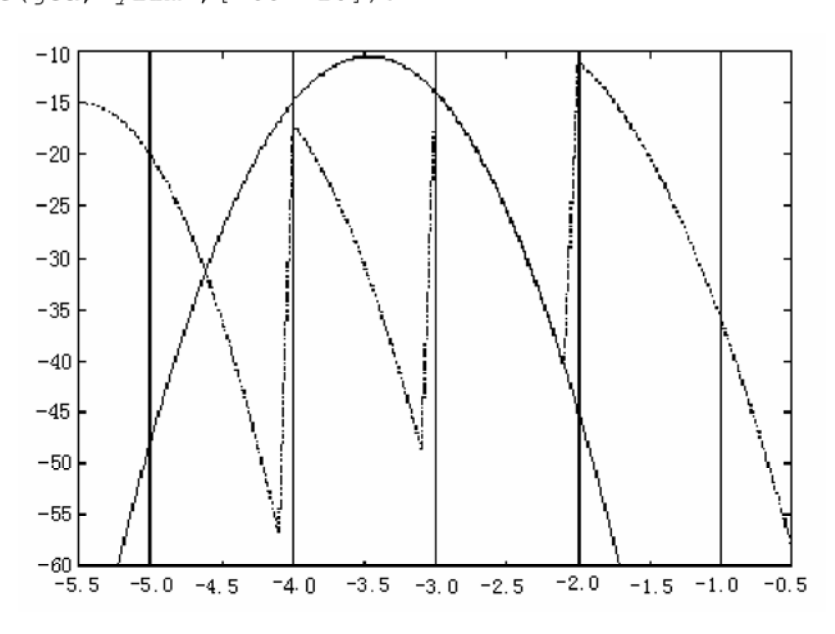


图 3.2.1　函数图形、节点及多项式的第三段

注意，这里函数中所说的节点序列 breaks 和以后函数中所讲的节点序列 knots 是有区别的，breaks 中的元素不能重复，且必须按严格递增次序排列，而 knots 则没有这些限制。所以有时为了区别起见，也称节点序列 breaks 为插值点序列，请读者留心。

下面介绍样条工具箱中提供的构造和操作 pp 形式样条函数的函数(见表 3.2.1)。

表 3.2.1 关于 pp 形式样条函数的函数

函 数 名	函 数 功 能
pp2sp	将 pp 形式的样条函数转化为 B 形式的样条函数
ppbrk	分解 pp 形式的样条函数
pplst	工具箱中可操作 pp 形式样条函数的函数列表
ppmak	构造 pp 形式的样条函数
pprfn	在 pp 形式样条函数中插入额外的节点重新构造样条函数
ppual	评价 pp 形式的样条函数

1 pp2sp

功能：将 PP 形式的样条函数转化为 B 形式的样条函数。

格式：sp=pp2sp(pp[, sconds])

说明：将 pp 所代表的 PP 形式的样条函数转化为 sp 所代表的 B 形式的样条函数；sconds 为一个向量，包含各个节点的光滑条件数；若忽略 sconds 参数，函数按缺省情况处理。

举例：
```
x=0:0.2:pi;
sconds=sin(x);
pp=csape(x,sconds);
sp=pp2sp(pp);
```

2 ppbrk

功能：分解 PP 形式的样条函数。

格式：[breaks, coefs, l, k, d]=ppbrk{pp}

out1=ppbrk(pp, part)

pj=ppbrk(pp, j)

pc=ppbrk(pp, [a, b])

说明：第一种格式下返回的是pp所代表的PP形式的样条函数的节点序列breaks，参数矩阵coefs，分段多项式段数L，样条函数的阶数K及维数D。第二种格式下返回的是由字符串类型参数part指定的部分，参数part可为下列字符串或字符串的第一个字母：‘breaks’，‘coefs’，‘pieces’或‘l’，‘order’或‘k’，‘dimension’，‘interval’。第三种格式下返回的是PP形式的样条函数的第j段(j为整数)。第四种格式下返回的是PP形式的样条函数在区间[a, b]内部的部分。

举例：
```
x=0:0.2:pi;
sconds=sin(x);
pp=csape(x,sconds);
[breaks,coefs,l,k,d]=ppbrk(pp);
out1=ppbrk(pp,  reaks? ;,
pj=ppbrk(pp,4);
pc=ppbrk(pp,[0.2:0.8];
```

3 pplst

功能：工具箱中可操作 PP 形式样条函数的函数列表。

格式：pplst

说明：直接调用，显示所有能够操作 PP 形式样条函数的函数清单，并有简略说明。其显示结果包含如下函数：

```
pp = ppmak(breaks,coefs[,d])
fnplt(pp[,{symbol,interv,linewidth}])
v = fnval(pp,x)
dpp =  fnder(pp[,m])
ipp =  fnint(pp[,ippa])
jump = fnjmp(pp,x)
fncmb(pp1,sc1[,pp2[,sc2]])
fn = fn2fm(pp,form)
sp = fn2fm(pp,'BB',sconds)
item = fnbrk(pp,part)
pj = fnbrk(pp,j)
pc = fnbrk(pp,[a b])
pq = fnrfn(pp,breaks)
```

4 ppmak

功能：构造 PP 形式的样条函数。

格式：ppmak

ppmak(breaks, coefs)

pp = ppmak(breaks, coefs, d)

说明：在给定节点序列和局部多项式参数矩阵的前提下，函数 ppmak 能用最少的信息构造 PP 形式的样条函数，所构造样条的详细信息可用命令 fnbrk 返回，并且存储样条信息的数据结构易被修改而不会影响其他函数的使用。调用 ppmak 时所产生的确切动作依赖于输入是单变元还是多变元，即 breaks 是一个插值点序列还是一个插值点阵列。

如果 breaks 是一个不减的插值点序列，且第一个元素和最后一个元素不等，则认为所要构造的样条是单变元的样条函数。它的各个部分按如下方式确定。

1. 样条的基本区间为 [breaks(1), breaks(l+1)]，多项式片段数 l = length(breaks □1)。
2. 目标样条函数的阶数k和维数d可以推断为：

a. 如果维数d没有明确给出，则coefs(:, i*k+j)被认为是包含第(i+1)个多项式片段的j个参数，且按降幂排列，即第一个参数为最高次幂的系数，最后一个参数为常数项，维数d由[d, kl] = size(coefs)获得，阶数k由k = fix(kl/l)获得；

b. 如果维数d明确给出，则k = cols(coefs)。

如果 breaks 是形如[BREAKS1, ..., BREAKSm]的阵列，其中 BREAKSi（i =1, ..., m）为插值点序列，则认为所要构造的样条是一个 m 变元的张量积样条函数。这种情况下，期望参数阵列 coefs 的大小为[d, lk]，这就定义了 d 的大小，其中 lk:= [l1*k1, ..., lm*km]，且对于所有 i，li = length(BREAKS{i}。

举例：
```
pp=ppmak([1,3,4],[1,2,5,6;3,4,7,8]);
pp=ppmak([1,3,4],[1,2;3,4;5,6;7,8]);
```
上述两种格式等价，一般Matlab内部使用第二种格式。

5 pprfn

功能：在 PP 形式样条函数中插入额外的节点重新构造样条函数。

格式：ppout = pprfn(pp, breaks)

说明：该函数在原样条函数 pp 中按大小顺序插入新的节点序列 breaks，并返回新的 PP 形式的样条函数。

如果原样条函数 pp 是 m 变元的张量积样条函数，则期望 breaks 是有 m 列的阵列。

举例：
```
x=0:0.2:pi;
sconds=sin(x);
pp=csape(x,sconds);
ppout=pprfn(pp,[0.3,0.7]);
```

6 ppual

功能：评价 PP 形式的样条函数。

格式：v=ppual(pp, xx[, left])

说明：返回 PP 形式的样条函数 pp 在向量 xx 中各点的函数值。如果第三个可选参数给出，意味着函数是左连续的。

如果 pp 是维数为 d 的单变元的样条函数，并且[m, n] == size(xx)，则输出为大小为[d*m, n]的矩阵。

如果 pp 是维数为 d 的 m 变元的样条函数，m>1，则输出如下大小的矩阵：

[d*m, n]，如果 xx 的大小为size [m,n]；

[d,n1,...,nm]，如果 d>1且 xx 为 {X1,...,Xm}；

[n1,...,nm]，　如果 d==1 且 xx 为 {X1,...,Xm}。

举例：
```
x=0:0.2:pi;
sconds=sin(x);
pp=csape(x,sconds);
v=ppual(pp,[0.1,0.8,1.2]);
```

3.3　B 形式样条函数的构造及使用

所谓 B 形式的样条函数，就是以 B 样条函数为基底，由 B 样条函数的线性组合来表示的样条函数。在高次多项式的计算中，选择稳定的计算公式是很关键的。因此对于固定分割、固定次数样条的全体形成的线性空间，问题就归结为如何选择基底。目前有三类基底在实际上受到人们的重视，一种是截断幂函数基底，一种是主基样条，另一种便是本节要介绍的 B 样条函数基底。截断幂函数基底主要用于理论分析，它们构成的系数矩阵是高度病态的，所以不适合用于数值计算。主基样条主要用于低次情况，如二次、三次样条的计算，对高次也不方便。这两组基底都不具有严格的误差局部化的性质。尽管主基样条近处小的扰动对远处影响很快衰减，但严格来说，相应的影响矩阵不是稀疏的，而且衰减效应随着多项式次数的增加而减小。截断幂函数基底一旦在某一处出现误差，就以幂次速度向远处单向传播，严重破坏计算稳定，而且这两组基底在处理具有重节点的样条时也不方便。B 样条函数则完全具有局部支柱性质，且能统一处理重节点，从而把样条函数的概念在理论上与分段多项式统一起来。

3.3.1　预备知识

单位阶跃函数的定义为：

$$\mu(x)=\begin{cases}1 & x\geq 0\\ 0 & x<0\end{cases}$$

显然函数 $\mu(x)$是分段常量，$x=0$是间断点，并在$x=0$处右连续。

截断幂函数的定义为

$$x_+^n=x^n\cdot\mu(x)=\begin{cases}x^n & x\geq 0\\ 0 & x<0\end{cases}$$

当 $n>0$ 时，$x_+{}^n$ 是连续的，且有

$$Dx_+^n=nx_+^{n-1},\Lambda,D^{n-1}x_+^n=n!\,x_+,D^nx_+^n=n!\,\mu(x)$$

式中

$$D=\frac{d}{dx}$$

为微分算子。显然函数 $x_+{}^n$ 是区间$(-\infty,+\infty)$上亏数为 1 的 n 次样条函数，它在仅有的一个内节点 $x=0$ 处取导数直到 n ? 1 阶都连续，而 n 阶导数是间断的。

当 x_i 是$[a,b]$上划分 Δ 的一个内节点时，函数$(x-x_i)_+{}^n$ 可以看作是$[a,b]$上亏数为 1 的 n 次样条函数。

定义 **4**　在区间$[a,b]$给定一个划分

$$\Delta：a = x_0 < x_1 < \Lambda < x_n = b$$

将满足定义 3 的样条函数的全体构成的集合，通称为 n 次样条函数空间，记为 $s_{n,v}(\Delta)$。

如果 $s_n(x)$为 $s_{n,v}(\Delta)$中的任一函数，α 为任一常数，显然有 $s_n(x)\in s_{n,v}(\Delta)$；另外，如果 $s_n(x)$，$r_n(x)$为 $s_{n,v}(\Delta)$的任两个函数，则 $s_n(x)+r_n(x)\in s_{n,v}(\Delta)$，这样便说明了 $s_{n,v}(\Delta)$是一个线性空间。

根据定义 3，空间 $s_{n,v}(\Delta)(v=0，1，2，\cdots)$有如下的包含关系：

$$s_{n,0}(\Delta)\in s_{n,1}(\Delta)\in s_{n,2}(\Delta)\in\Lambda$$

于是函数族

$$\begin{cases} x^k & k = 0,1,\Lambda\ ,n \\ (x - x_i)_+^j & j = n - \nu + 1, n - \nu + 2,\Lambda\ ,n; i = 1,2,\Lambda\ ,N - 1 \end{cases}$$

中的每一个函数均在 $s_{n,v}(\Delta)$之内，且有以下引理和定理。

引理：　函数族中的 $n+1+v(N-1)$个函数是线性无关的。

定理：　样条函数空间 $s_{n,v}(\Delta)$是 $n+1+v(N-1)$维的线性空间。

3.3.2　B 样条函数（Basic spline function）

定义 **5**　在实轴上取节点序列

$$\Lambda < x_{-n} < \Lambda < x_0 < \Lambda < x_N < \Lambda < x_{N+n} < \Lambda$$

在截断幂函数$(t-x)_+^n$中将 x 看作参数，则关于 $t=x_i, x_{i+1}, ..., x_{i+n+1}$ 作函数$(t-x)_+^n$的 $n+1$ 阶差商

$$[x_i, x_{i+1},\Lambda\ ,x_{i+n+1}](t-x)_+^n$$

称此差商是以 x 为变量的 B 样条函数，并称

$$B_{i;n+1}(x) = (x_{i+n+1} - x_i)[x_i, x_{i+1},\Lambda\ ,x_{i+n+1}](t-x)_+^n$$

为第 i 个 $n+1$ 阶规范 B 样条函数。

根据差商与各节点处函数值之间的关系式得

$$B_{i;n+1}(x) = (x_{i+n+1} - x_i)\sum_{k=i}^{i+n+1}\frac{(x_k - x)_+^n}{\omega'_{i,n+1}(x_k)}$$

其中

$$\omega_{i,n+1}(x) = \prod_{j=i}^{i+n+1}(x - x_j)$$

$$\omega'_{i,n+1}(x_k)=\prod_{\substack{j=i\\ j\neq k}}^{i+n+1}(x_k-x_j)$$

根据如上表达式，明显可以看出 $B_{i;n+1}(x)$是分段 n 次多项式。如果取

$$x_{-n}=\Lambda=x_{-1}=a=x_0<\Lambda<x_N=b=x_{N+1}\Lambda=x_{N+n}$$

时，称 $B_{-n;n+1}(x)$，$B_{-n+1;n+1}(x)$，…，$B_{-1;n+1}(x)$分别是以 $x=a$ 为 n+1 重，n 重，…，二重节点的 n+1 阶 B 样条函数，$B_{N-1;n+1}(x)$，$B_{N-1;n+1}(x)$，…，$B_{N-n;n+1}(x)$分别是以 $x=b$ 为 n+1 重，n 重，…，二重节点的 n+1 阶 B 样条函数。其余的 $B_{i;n+1}(x)$ (i=0,1,...,$N-n-1$)均为单节点的 $n+1$阶样条函数。带有重节点的 B 样条函数可按 n+1 阶差商的极限情况来定义。例如 $B_{-n;n+1}(x)$可定义为当 $x_{-n}, x_{-n+1}, ..., x_{-1}$ 均趋近于 a 时，$(x_1-x_0)[x_{-n}, x_{-n+1}, ..., x_1](t-x)_+^n$ 的极限。

下面介绍 3 个低阶的 B 样条函数。

1　一阶 B 样条函数

第 i 个一阶(零次)B 样条函数为

$$\begin{aligned}B_{i;1}(x)&=(x_{i+1}-x_i)[x_i,x_{i+1}](t-x)_+^0\\&=\left(\frac{(x_{i+1}-x)_+^0}{x_{i+1}-x_i}-\frac{(x_i-x)_+^0}{x_{i+1}-x_i}\right)(x_{i+1}-x_i)\\&=\begin{cases}1 & x_i\le x<x_{i+1}\\ 0 & \text{其他}\end{cases}\end{aligned}$$

其图形为图 3.3.1。

2　二阶 B 样条函数

第 i 个二阶(一次)B 样条函数为

$$\begin{aligned}B_{i;1}(x)&=(x_{i+2}-x_i)[x_i,x_{i+1},x_{i+2}](t-x)_+^1\\&=\begin{cases}\dfrac{x-x_i}{x_{i+1}-x_i} & x_i\le x<x_{i+1}\\ \dfrac{x_{i+2}-x}{x_{i+2}-x_i} & x_{i+1}\le x<x_{i+2}\\ 0 & \text{其 他}\end{cases}\end{aligned}$$

其图形为图 3.3.2。

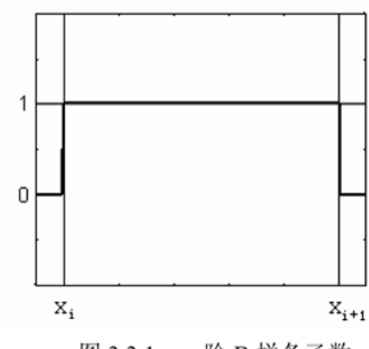


图 3.3.1 一阶 B 样条函数

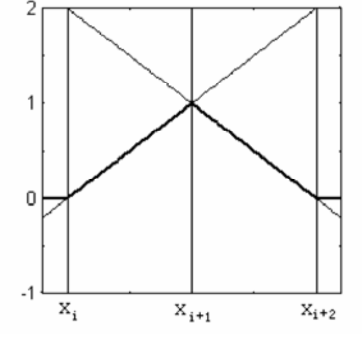


图 3.3.2 二阶 B 样条函数

3 三阶 B 样条函数

第 i 个三阶(二次)B 样条函数为

$$B_{i;1}(x)=(x_{i+3}-x_i)[x_i,x_{i+1},x_{i+2},x_{i+3}](t-x)_+^2$$
$$=\begin{cases}\dfrac{(x-x_i)^2}{(x_{i+1}-x_i)(x_{i+2}-x_i)} & x_i\le x<x_{i+1}\\ \dfrac{(x-x_{i+2})^2}{(x_{i+1}-x_i)(x_{i+2}-x_i)}-\dfrac{(x_{i+2}-x_{i+1})(x-x_{i+1})}{(x_{i+2}-x_{i+1})(x_{i+3}-x_{i+1})} & x_{i+1}\le x<x_{i+2}\\ \dfrac{(x_{i+3}-x)^2}{(x_{i+3}-x_{i+1})(x_{i+3}-x_{i+2})} & x_{i+2}\le x<x_{i+3}\\ 0 & \text{其 他}\end{cases}$$

其图形为图 3.3.3。

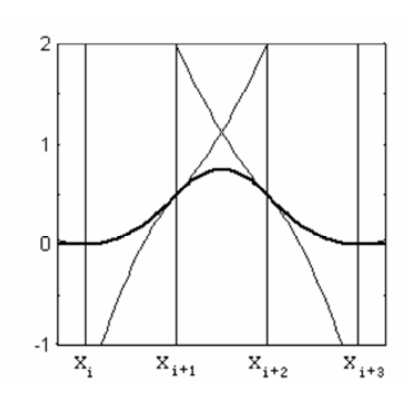


图 3.3.3 三阶 B 样条函数

3.3.3 B 样条函数的性质

通过 B 样条函数的定义及其表达式，不难得到 B 样条函数的以下性质。这些性质的证

明过程本书不详细给出，读者可以自行参考有关文献或推导。

1　递推关系式性质

$$B_{i,k+1}(x)=\frac{x-x_i}{x_{i+k}-x_i}B_{i,k}(x)+\frac{x_{i+k+1}-x}{x_{i+k+1}-x_{i+1}}B_{i+1,k}(x)$$

其中，　$k=1,2,...,n$；　$i=1,2,...N-1$。

从这个表达式中可以看出，k 次 B 样条函数 $B_{i,k+1}(x)$可由两个 $k-1$ 次的 B 样条函数 $B_{i,k}(x)$和 $B_{i+1,k}(x)$表示出来，若加上

$$B_{i,1}(x)=\begin{cases}1 & x_i\le x<x_{i+1}\\ 0 & 其他\end{cases}$$

则可以得到 B 样条函数的递推定义。

2　局部正支撑性质

B 样条函数 $B_{i,k+1}(x)(i=-n, ..., N-1)$有下列性质：

$$B_{i,n+1}(x)\begin{cases}>0 & x_i\le x<x_{i+1}\\ \equiv 0 & 其他\end{cases}$$

这用数学归纳法证明即可。由于 B 样条函数恒为非负，而且 $n+1$ 阶(n 次)B 样条函数只在 n 个子区间上非零，则其具有局部正支撑性。

3　规范性(又称权性)

对于 $x_{-n}\leqslant x<x_{N+n}$ 的 B 样条函数有

$$\sum_{i=-n}^{N+n}B_{i,n+1}(x)\equiv 1$$

这个性质可以直接由 B 样条函数的差商的定义证明。

4　线性无关性

n 次 B 样条函数 $B_{i,n+1}(x)(i=-n, ..., N+n)$是线性无关的，即对于常数 $c_i(i=-n, ..., N+n)$，

$$\sum_{i=-n}^{N+n}c_iB_{i,n+1}(x)\equiv 0$$

成立的充分条件是 $c_i=0$（$(i=-n, ..., N+n)$）。从这个性质知道，n 次 B 样条函数组 $B_{i,n+1}(x)$（$(i=-n, ..., N+n)$）构成 n 次多项式样条函数空间的一组基底。

5　微分性质

B 样条函数 $B_{i,n+1}(x)$的导数可用下式来表达

$$B'_{i,n+1}(x)=\frac{n}{x_{i+n}-x_i}B_{i,n}(x)-\frac{n}{x_{i+n+1}-x_{i+1}}B_{i,n+1}(x)$$

这可用 B 样条函数的差商定义证明。由此可以看出，B 样条函数的导数可由两个低一次的 B 样条函数的线性组合来表示。

3.3.4　工具箱中关于 B 形式样条函数的函数

在 Matlab 提供的样条工具箱中，B 形式的样条函数 f 是由它的不减的节点序列 t 和 B 样条函数系数阵列 a 来唯一表达的。

已知阵列 a 的大小为[d, n] = size(a)和节点序列 t 的长度为 length(t)，则可知道样条函数 f 的阶数 k = length(t)−n，这就意味着分段多项式样条函数的次数<k，例如三次样条函数就是一个四阶样条。

确切地说，B 形式的样条函数可由 t, a, n 和 k 四部分构成，且可以表示为

$$f:=\sum_{i=1}^{n}B_{ik}*a(:,i)$$

其中

$$B_{ik}=B(\cdot\,|\,t(i:i+k))$$

表示给定节点序列 $t(i)$, …, $t(i+k)$下的第 i 个 k 阶 B 样条函数，它的基本区间是[$t(1)$，$t(n+k)$]。注意 B 样条函数在它的基本区间外恒为零，这同 pp 形式的样条函数是不同的。

B 样条函数是构造 B 形式样条函数的基本单元。下面给出一个以[0 1.5 2.3 4 5]为节点序列的四阶 B 样条函数的图形(见图 3.3.4)，同时给出构成样条函数的分段多项式的图形。命令如下：

```
bspline([0 1.5 2.3 4 5])
```

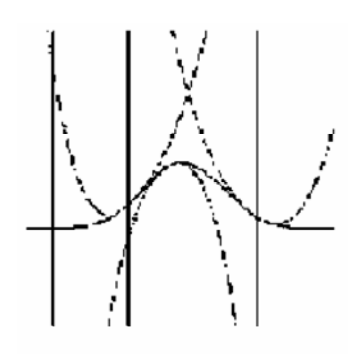


图 3.3.4　四阶 B 样条函数

概括地说，以 t(i)≤…≤t(i+k)为节点序列的 B 样条函数在区间[(t(i)，t(i + k))]内为正，

区间外为零。这些节点可以一致，并且节点的重数决定了该点的连续性条件，规则是：

节点重数 ＋ 连续性条件个数 ＝ 阶数

例如对于三阶 B 样条函数，单节点意味着两个连续性条件，比如函数连续和一阶导数连续；双节点则意味着只有一个连续性条件，即函数连续；三重节点就意味着没有连续性条件，即函数也可以为不连续的。 这对节点的选择有很大的影响。 假定 s 为区间[a，b]上的以 $\xi_2<\cdots<\xi_l$ 为插值点序列的 k 阶样条函数，进一步假定在插值点 ξ_i 处样条函数 s 满足 u_i 个光滑条件，即

$$jump_{\xi i}D^j s(\xi_i +)-D^j s(\xi_i -)=0\text{，}\ 0\le j<\mu_i \qquad i=2,\Lambda\ ,l$$

那么合理的节点序列 t 应该包含 $k\text{-}u_i$ 个插值点 ξ_i，i=2,...,l，并且 t 还应包含两个端点 a 和 b 各 k 个。这样构造的节点序列 t，就可以将样条函数 s 用 B 样条函数唯一地表示出来。例如，想在区间[1, 3]上以 1.5、1.8、2.6 为插值点构造一个三次样条函数，同时要求满足函数存在二阶的连续导数，则节点序列可以选择

t = [1, 1, 1, 1, 1.5, 1.8, 2.6, 3, 3, 3, 3]

它由如下命令得到

`augknt([1, 1.5, 1.8, 2.6, 3], 4)`(这个函数在第五节介绍)。

如果想在插值点 1.8 处为一个角，即函数在该点的一阶导数不连续，则可用节点序列

t = [1, 1, 1, 1, 1.5, 1.8, 1.8, 1.8, 2.6, 3, 3, 3, 3];

它由如下命令得到：

```
t = augknt([1, 1.5, 1.8, 2.6, 3], 4, [1, 3, 1] );
```

图 3.3.5 展示包含不同节点重数的三次 B 样条函数的图形，该图形可在 Matlab 中用以下命令得到：

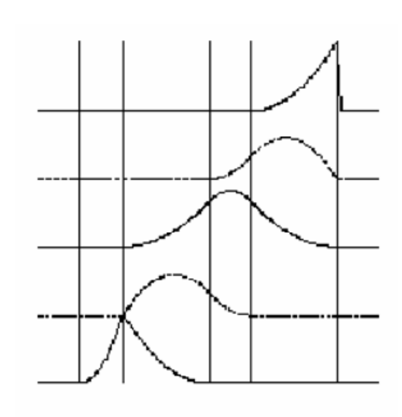


图 3.3.5　包含不同节点重数的三次 B 样条函数

```
x=[-10:70]/10;
c=spcol(t,3,x);
[l,m]=size(c);
c=c+ones(l,1)*[0:m-1];
axis([-1 7 0 m]);
```

```
for tt=t, plot([tt tt],[0 m],? ? , end
plot(x,c)
```

更详细的解释见 spalldem。

通常，样条函数是由一些信息来构造的，比如函数值、导数值或者常微分方程的近似解等。但是，也有可能通过拼装而成。Matlab 中提供了这样的函数 spmak，只要给定节点序列和系数阵列就可以得到 B 形式的样条函数。例如

```
sp = spmak(1:10,3:8)
```

提供了节点序列 1 : 10 和系数阵列 3 : 8，这样就可以得到阶数 $k = 4(= 10-6)$。通过命令

```
fnbrk (sp)
```

可以得到详细信息如下：

```
knots(1:n+k)
1 2 3 4 5 6 7 8 9 10
coefficients(d,n)
3 4 5 6 7 8
number n of coefficients
6
order k
4
dimension d of target
1
```

但是样条工具箱的最大特点就是不需要读者了解这些详细信息，只要使用工具箱提供的函数和命令就可对包含在 sp 中的样条函数进行评价、微分、积分和转换等操作。

下面介绍样条工具箱中提供的构造和操作 B 形式样条函数的函数。

表 3.3.1 工具箱中关于 B 形式样条函数的函数

函 数 名	函 数 功 能
sp2bb	将 B 形式的样条函数转化为 BB 形式的样条函数
sp2pp	将 B 形式的样条函数转化为 PP 形式的样条函数
spap2	最小均方误差的样条逼近
spapi	样条插值
spaps	构造光滑的样条函数
spbrk	分解样条函数
spcol	产生 B 样条函数的配置矩阵
splst	工具箱中能够操作 B 形式样条函数的函数的列表
spmak	构造 B 形式的样条函数
sprfn	在样条函数的节点序列中插入额外的节点并返回新的样条函数
spcrv	产生一个样条曲线
spval	求值

1 sp2bb

功能：将 B 形式的样条函数转化为 BB 形式的样条函数。

格式：PP = SP2BB(SPLINE)

说明：通过节点插入函数使原 B 形式的样条函数中的每个节点的重数变为最大，即样条函数的阶数，然后利用新的节点序列得到新的样条函数，即 BB 形式的样条函数。基本过程如下：

```
[xi,m] = knt2brk(spbrk(spline,'knots'));
spline = sprfn(spline, brk2knt(xi,?
subplus(spbrk(spline,'order') - m)));
spline(1) = 12;
```

举例：

```
[xx,yy]=titanium;
pick=[1 5 11 21  27 29 31 33 35 40 45 49];
tau=xx(pick);
y=yy(pick);
n=length(tau);
dl=tau(2)-tau(1);
dr=tau(n)-tau(n-1);
t=[tau(1)-dl*[2 1] tau tau(n)+dr*[1 2]];
sp=spapi(t,tau,y);
pp = sp2bb(sp);
```

2 sp2pp

功能：将 B 形式的样条函数转化为 pp 形式的样条函数。

格式：pp=sp2pp(spline)

说明：在原区间上，将 B 形式的样条函数转化为 pp 形式的样条函数。

举例：

```
[xx,yy]=titanium;
pick=[1 5 11 21  27 29 31 33 35 40 45 49];
tau=xx(pick);
y=yy(pick);
n=length(tau);
dl=tau(2)-tau(1);
dr=tau(n)-tau(n-1);
t=[tau(1)-dl*[2 1] tau tau(n)+dr*[1 2]];
sp=spapi(t,tau,y);
pp = sp2pp(sp);
```

3 spap2

功能：最小均方误差的样条逼近。

格式：sp= spap2(knots, k, x, y[, w])

说明：返回通过节点序列 knots 的 k 次样条函数 f，并使之在最小均方误差的意义下满足条件 y=f(x)，即使

$$\sum_j W(j)(Y(j)-f(X(j)))^2$$

为最小。缺省情况下权重 W 为 ONES(size(x))，但另一种更好的选择是使 W = ([dx ; 0]+[0 ; dx]).'/2，其中 dx = diff(x(:))。当在横轴上满足 Schoenberg-Whitney 条件，即满足

knots (j) < x(j) < knots (j+k)，j=1: length(x)=length(knots)-k

时，所得的样条函数是唯一的；并且在横轴上必须有一些 j 满足该条件，否则函数没有返回值。同样该函数可以用于构造张量积样条函数，此时要注意各个输入参数的匹配问题。

举例：

```
[xx,yy]=titanium;
k=3;
l=6;
breaks=x(1)+[0:l]*(x(length(x))-x(1))/l;
knots=augknt(breaks,k);
spline=spap2(knots,k,x,y);
```

4 spapi

功能：样条插值。

格式：sp=spapi(knots, x, y)

说明：返回以 konts 为节点序列，以 k=length(knots)−length(x)为阶的满足条件

y = f(x)

的样条函数 sp。同样该函数可以用于构造张量积样条函数，此时要注意各个输入参数的匹配问题。

举例：

```
[xx,yy]=titanium;
pick=[1 5 11 21  27 29 31 33 35 40 45 49];
tau=xx(pick);
y=yy(pick);
n=length(tau);
dl=tau(2)-tau(1);
dr=tau(n)-tau(n-1);
t=[tau(1)-dl*[2 1] tau tau(n)+dr*[1 2]];
sp=spapi(t,tau,y);
```

5 spaps

功能：构造光滑的样条函数。

格式：[sp, values]=spaps(x, y, tol{, w, m})

说明：返回使 F(D^M f) := integral { (D^M f(t))^2 : x(1)<=t<=x(n) }为最小的光滑函数 f 的 B 形式的样条函数 sp 和序列 x 处的函数值，并使函数 f 满足条件

$$E(f)=\sum_j W(j)(Y(j)-f(X(j)))^2 \le TOL$$

在缺省情况下，权重 W 使 $E(f)$近似于 F(y- f)。若为三次样条函数则缺省 M=2，也可选择 M=1(线性)或 M=3(五次)。若序列 x 为非增的，则要将 x, y 重新排序。同样该函数可以用于构造张量积样条函数。当 x 是一个长度为 length(x) = r 的单元阵列(其每个单元为一个向量)时，y 则应该提供相应的节点数据，且对于标量数据，y 的大小为 size [length(x{i}): i=1: r]，对于 d 维向量数据，y 的大小为 size [d, [length(X{i}): i=1: r]]。在这种情况下，M 是一个整数或一个 大小为 r 的向量(其元素属于点集[1, 2, 3])，W 则是以向量 W(i)为元素的长度为 r 的阵列。

举例：(1)　单变元

```
x=0:0.2:pi;
y=sin(x);
tol=1;
[sp,values]=spaps(x,y,tol);
```

(2)　多变元

```
w = ones(size(x));
w([1 end]) = 100;
sp = spaps(x,y,1.e-2,w,3);
```

6　spbrk

功能：分解样条函数。

格式：[out1, coefs, n, k, d]=spbrk(sp)

out1=spbrk(sp, part)

spbrk(sp)

说明：第一种格式下返回的是sp所代表的B形式的样条函数的节点序列out1，参数向量coefs及其长度n，样条函数的阶数k及维数d。第二种格式下返回的是由字符串类型参数part指定的部分，参数part可为下列字符串或字符串的第一个字母：'knots'，'t'，'coefs'，'number'，'order'，'dimension'，'interval'。第三种格式下返回的内容同第一种格式。

举例：

```
[xx,yy]=titanium;
pick=[1 5 11 21 27 29 31 33 35 40 45 49];
tau=xx(pick);
y=yy(pick);
n=length(tau);
dl=tau(2)-tau(1);
dr=tau(n)-tau(n-1);
t=[tau(1)-dl*[2 1] tau tau(n)+dr*[1 2]];
sp=spapi(t,tau,y);
```

```
spbrk(sp);
out1=spbrk(sp,  rder? ;
[out1,coefs,n,k,d]=spbrk(sp);
```

7 spcol

功能：产生 B 样条函数的配置矩阵。

格式：spcol(knots, k, tau)

colloc = spcol(knots, k, tau, arg1, arg2)

说明：函数 spcol 构造矩阵

$$colloc = \left(D^{m(j)} B_j\left(tau(i)\right)\right)$$

其中 B_j 是以 knots 为节点序列的第 j 个 k 阶 B 样条函数，tau 是一个不减的点序列，并且 m = knt2ml(tau),即

m(i) := #{ j<i : TAU(j) = TAU(i) }

如果一个可选参数是字符串‘slvblk’的前两个字母，则函数返回函数 slvblk 所需要的块对角线形式的矩阵；如果一个可选参数是字符串‘sprase’的前两个字母，则函数返回稀疏矩阵；如果一个可选参数是字符串‘noderiv’的前两个字母，则函数忽略所有节点的重数，即对于所有的 i，使 m(i) = 1。

举例：

```
spcol([1:6],3,.1+[2:4])
ans =
   0.5900 0.0050 0
   0.4050 0.5900 0.0050
   0 0.4050 0.5900
bkbrk(spcol([1:6],3,.1+[2:4],'sl'))
ans =
   block 1 has 2 row(s)
   0.5900 0.0050 0
   0.4050 0.5900 0.0050
   next block is shifted over 1 column(s)
   block 2 has 1 row(s)
   0.4050 0.5900 0.0050
   next block is shifted over 2 column(s)
```

8 splst

功能：工具箱中能够操作 B 形式样条函数的函数的列表。

格式：splst

说明：直接调用，显示所有能够操作 B 形式样条函数的函数清单，并有简略说明。其显示结果包含如下函数：

```
sp = spmak(knots, coefs);
sp = spapi(t,tau,ftau) ;
sp = spap2(t,k,tau,ftau,w);
sp = spaps(x,y,tol,m,w)  ;
fnplt(sp,symbol,interv,linewidth);
values = fnval(sp,x) ;
dsp = fnder(sp,m)  ;
isp = fnint(sp,ispa) ;
sp = fncmb(sp1,sc1,sp2,sc2);
item = fnbrk(sp,part)  ;
jump = fnjmp(sp,x) ;
spr = fnrfn(sp,knots);
fm = fn2fm(sp,form) ;
c = spcol(t,k,tau,form,noderiv)  ;
values = csapi(x,y,xx);
pp = csapi(x,y)  ;
pp = csape(x,y,conds,valconds) ;
pp = cscvn(points)  ;
values = csaps(x,y,p,xx);
[curve=] spcrv(c[,k,symbol,maxpnt]);
[newbrk,distfn] = newknt(pp,newl)  ;
xi = optknt(tau,k)  ;
[augknot,addl] = augknt(knots,k);
tstar=aveknt(t,k);
t = brk2knt(breaks,mults);
[xi,m] = knt2brk(t);
[m,t] = knt2mlt(t);
pointer = sorted(knots, points) ;
[v,b] = sprpp(tx,a)  ;
[v,b] = splpp(tx,a)  ;
[nb,rows,ncols,last,blocks] = bkbrk(blokmat);
```

9　spmak

功能：构造 B 形式的样条函数。

格式：spline=spmak(knots, coefs)

说明：在 Matlab 中,B 形式的样条函数是由它的不减的节点序列 knots 和 B 样条函数系数序列 coefs 来表示的，其形式如下：

$$spline = \sum_{i=1}^{n} B(i,k) * coefs(i)$$

其中

$$n = length(coefs)$$

$$n + k = length(knots)$$

B(i,k)表示第 i 个 k 阶 B 样条函数。

在函数 spmak 中，knots 为节点序列，coefs 为系数矩阵，其列数 n 代表样条函数由 n 个 B 样条函数组成，行数 d 代表维数，所构造的样条函数的阶数为 k=length(knots)−n。同样该函数可以用于构造张量积样条函数。如果 knots 是 m 维的阵列，则 coefs 是 m 或 m−1 维的阵列。

举例：`spline=spmak(0:10,0:7)`

10 sprfn

功能：在样条函数的节点序列中插入额外的节点，并返回新的样条函数。

格式：spnew =sprfn(sp, addknots)

说明：在原样条函数 sp 的节点序列中插入新的节点 addknots，若插入若干节点时，则 addknots 为一个节点序列。然后函数以新的节点序列构造新的样条函数。但是没有节点的重数超过样条的阶数。同样该函数可以用于张量积样条函数，如果 sp 是 m 变元的样条，则 addknots 应是一个包含 m 个单元的阵列。

举例：`spnew=sprfn(sp,0:7)`

11 spcrv

功能：产生一个样条曲线。

格式：curve = spcrv(x, k, maxpnt)

spcrv(x)

说明：该函数为以 c 为参数阵列的 k 阶样条曲线 f 提供了密集的点序列 f(tt)。确切地，曲线 f 可以记为

$$f : t \to \sum_{j=1}^{n} B(t \mid -k/2 \mid j, \Lambda\ , j+k) * c(j), \qquad \frac{k}{2} \le t \le n + \frac{k}{2}$$

其中 B(• |a,⋯,z)代表以 a, ⋯,z 为节点的 B 样条，n 是样条函数参数阵列 c 的列数，即[d, n] = size（c）。k 的缺省值为 4，序列 tt 的缺省元素的最大个数为 100。

举例：
```
points = [0 0 1 1 0 -1 -1 0 0 ;
          0 0 0 1 2 1 0 -1 -2];
```

```
plot(points(1,:),points(2,:),':')
values = spcrv(points,3);
plot(values(1,:),values(2,:));
```

12　spval

功能：求值。

格式：v = spval(sp,x)

说明：返回 B 形式样条函数在节点序列 x 处的值。

举例：

```
[xx,yy]=titanium;
pick=[1 5 11 21 27 29 31 33 35 40 45 49];
tau=xx(pick);
y=yy(pick);
n=length(tau);
dl=tau(2)-tau(1);
dr=tau(n)-tau(n-1);
t=[tau(1)-dl*[2 1] tau tau(n)+dr*[1 2]];
sp=spapi(t,tau,y); v=spval(sp,0:7)
```

3.4　张量积样条函数

二元样条函数的研究始于 20 世纪 60 年代，比单元的样条函数晚了十多年。由于多元函数本身在计算上的繁杂性及多维区域剖分的多样性，导致了研究上的困难重重，使之在最初的一段时间进展缓慢。不过，近许多年来引起了普遍的重视，研究学者增多，进展较快，成果不少。

目前，规则区域的二元样条函数的研究已经趋于成熟，非规则区域的逼近以及离散点的二元插值仍是研究的焦点之一。由于有限元和计算机辅助几何设计的大量应用，这方面也取得了相当的成果。

本节先着重描述一些简单的二元样条函数及其构成方法，然后介绍 Matlab 样条工具箱中关于二元样条函数的函数。

3.4.1　二元样条函数

首先介绍一些相关的概念。一般，设 $X=[a,b]$，$Y=[c,d]]$，令

$$X_n = span\{\phi_0,\phi_1,\Lambda\ ,\phi_n\}$$

$$Y_n = span\{\psi_0,\psi_1,\Lambda\ ,\psi_m\}$$

分别表示由 X，Y 中基函数

$$\{\phi_i\}(0 \le i \le n) 与 \{\psi_j\}(0 \le j \le m)$$

张成的子空间，它们的维数分别为 $n+1$ 和 $m+1$。

定义 **6** 记张量积为

$$X_n \otimes Y_m = span\{\phi_i(x)\psi_j(y); \quad 0 \le i \le n, 0 \le j \le m\}$$

上面定义表明张量积 $X_n \odot Y_m$ 表示从$\{\phi_0, \phi_1, ..., \phi_n\}$中任取 $\phi_i(x)$，从$\{\psi_0, \psi_1, ..., \psi_m\}$任取 $\psi_j(y)$的所有可乘积的线性组合的全体。显然这是 $C[a,b]\times C[c,d]$中的一个$(n+1)(m+1)$维的线性子空间，且

$$\Phi = \{\phi_i(x)\psi_j(y); 0 \le i \le n, 0 \le j \le m\}$$

是 $X_n \odot Y_m$ 的一组基，因此任一函数 $s \in X_n \odot Y_m$ 可以唯一表示为

$$s(x, y) = \sum_i \sum_j \alpha_{ij}\phi_i(x)\psi_j(y)$$

对于整数 $k\geqslant 0, l\geqslant 0$，用 $C^{k,l}[\Omega]$表示二元函数 $f(x,y)$在区域 Ω 上分别关于 x 及 y 具有直到 k 阶和 l 阶的连续偏导数(当然不大于 $k+l$ 阶的混合偏导数也是连续的)的总体集合。$C^k[\Omega]$表示二元函数 $f(x,y)$在区域 Ω 上只能保证分别关于 x 及 y 具有 k 阶的连续偏导数的总体集合。

1 双一次样条

对于给定的矩形区域

$$\Omega = \{(x, y); \quad a \le x \le b, c \le y \le d\}$$

引入乘积型剖分 $\Delta = \Delta_x \Delta_y$，其中

$$\Delta_x: \ a = x_0 < x_1 < \Lambda < x_n = b$$

$$\Delta_y: \ c = y_0 < y_1 < \Lambda < y_m = d$$

将区域 Ω 划分成 nm 个小的矩形域

$$\Omega_{i,j} = \{(x, y): x \in [x_i, x_{i+1}], y \in [y_j, y_{j+1}]; i = 0,1,\Lambda, n-1, y = 0,1,\Lambda, m-1\}$$

在这样的网状剖分上的所有点 (x_i, y_j)处的函数值记为 $f(x_i, y_j)=f_{i,j}$。当 j 固定，所得数据的插值一次样条函数可以表示为

$$s_{1,\Delta x} f = \sum_{i=0}^{n} f_{i,j} B_i(x)$$

其中 $B_i(x)$是 3.3 节中给出的二阶 B 样条函数。同理，当 i 固定，所得数据的插值一次样条函数可以表示为

$$s_{1,\Delta y} f = \sum_{j=0}^{m} f_{i,j} B_j(y)$$

于是对一次张量积插值样条函数可以表示为

$$s_{1,\Delta}(f;x,y) = \sum_{i=0}^{n}\sum_{j=0}^{m} f_{i,j} B_i(x) B_j(y)$$

由此可以得到二元双线性函数的基底函数，即二元双一次 B 样条函数的定义见图 3.4.1。

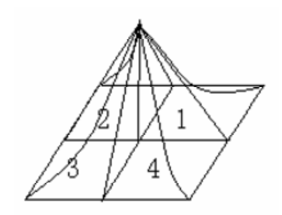


图 3.4.1　区域划分

$$\varphi_{i,j}(x,y)=\begin{cases}\dfrac{(x_{i+1}-x)}{(x_{i+1}-x_i)}\cdot\dfrac{(y_{j+1}-y)}{(y_{j+1}-y_j)}, & \text{区域1}\\[2ex] \dfrac{(x-x_{i-1})}{(x_i-x_{i-1})}\cdot\dfrac{(y_{j+1}-y)}{(y_{j+1}-y_j)}, & \text{区域2}\\[2ex] \dfrac{(x-x_{i-1})}{(x_i-x_{i-1})}\cdot\dfrac{(y-y_{j-1})}{(y_j-y_{j-1})}, & \text{区域3}\\[2ex] \dfrac{(x_{i+1}-x)}{(x_{i+1}-x_i)}\cdot\dfrac{(y-y_{j-1})}{(y_j-y_{j-1})}, & \text{区域4}\\[2ex] 0, & \text{其他处}\end{cases}$$

$$\varphi_{i,j}(x,y) = B_i(x) B_j(y)$$

如果被插函数 f 在区域 Ω 上 a−1 阶偏导数连续、a 阶偏导数在网格上分片连续、且在该区域内平方可积，则记 $f \in PC_{(2)}{}^{a}(\Omega)$。如果后一条件改为分片 a 阶偏导数按最大值一致有界，则记 $f \in PC^{a}(\Omega)$。下面的定理表明，双线性插值样条函数在 L^2 模意义下是二阶逼近的。

定理　如果

$f \in PC^{\alpha}(\Omega)$，则

$$\left\| f - s_{1,\Delta}(f;x,y) \right\|_2 \le \frac{1}{8}\left[h^2 \left\| f^{(2,0)}(x,y) \right\|_\infty + l^2 \left\| f^{(0,2)}(x,y) \right\|_\infty \right]$$

其中

$$h = \max_{0\le i\le n-1}(x_{i+1}-x_i),\quad l = \max_{0\le j\le m-1}(y_{j+1}-y_j)$$

$f^{(2,0)}(x,y)$及$f^{(0,2)}(x,y)$分别表示$f(x,y)$对于x和y的二阶偏导数。

2 双二次样条函数

对于矩形区域$\Omega=\{(x,y);a\leqslant x\leqslant b,c\leqslant y\leqslant d\}$作矩形剖分

$$\Delta_x：\ a=x_0<x_1<\Lambda<x_n=b$$

$$\Delta_y：\ c=y_0<y_1<\Lambda<y_m=d$$

在此划分的基础上，再作出由下列不等式规范的各小矩形

$$\Omega_{ij}:\begin{cases} x_{i-\frac{1}{2}}=\dfrac{x_i+x_{i-1}}{2}\leq x\leq\dfrac{x_i+x_{i+1}}{2}=x_{i+\frac{1}{2}} \\ y_{j-\frac{1}{2}}=\dfrac{y_j+y_{j-1}}{2}\leq y\leq\dfrac{y_j+y_{j+1}}{2}=y_{j+\frac{1}{2}} \end{cases}$$

式中：

$$j=1,2,\Lambda,m;\ i=1,2,\Lambda,n;\ x_{-1}=x_0;\ y_{-1}=y_0;\ x_{n+1}=x_n;\ y_{m+1}=y_m$$

如果函数$s_{2,\triangle}(x,y)$满足下列两个条件：

1) $s_{2,\Delta}(x,y)\in C^{1,1}[\Omega]$

2) 在每个Ω_{ij}上为双二次多项式

$$u_{ij}(x,y)=\sum_{k,l=0}^{2}\gamma_{k,l}^{ij}\left(x-x_{i-\frac{1}{2}}\right)^k\left(y-y_{j-\frac{1}{2}}\right)^l$$

式中j=1,2,...,m；i=1,2,...,n，则称$s_{2,\triangle}(x,y)$为双二次样条函数。

由定义和推理可以知道，$s_{2,\triangle}(x,y)$中共有 9+3n+3m+mn 个自由参数，即在此剖分网上双二次样条函数空间的维数为(n+3)(m+3)。如果考虑的是插值问题，网格节点处给出(n+1)(m+1)个函数值：

$$s_{2,\Delta}(x_i,y_j)=f_{ij};\ i=0,1,\Lambda,n;\ j=0,1,\Lambda,m$$

剩下的 2(n+m+4)个自由参数由边界条件确定。

与一维时类似，二维样条的边界条件可分为周期和非周期两大类。对于非周期边界，当边界上给定一阶或二阶发向导数时，还需给定角点处的二阶或四阶混合偏导数，它们分别为一类边界条件

$$\begin{cases} s_{2,x}(x_i,y_j)=f_{ij}^{1,0}, & i=0,n;j=0,\Lambda,m \\ s_{2,y}(x_i,y_j)=f_{ij}^{0,1}, & i=0,\Lambda,n;j=0,m \\ s_{2,xy}(x_i,y_j)=f_{ij}^{1,1}, & i=0,n;j=0,m \end{cases}$$

与二类边界条件

$$\begin{cases} \dfrac{\partial}{\partial x^2} s_2\left(x_i, y_j\right) = f_{ij}^{2,0}, & i = 0, n; j = 0, \Lambda\ , m \\ \dfrac{\partial}{\partial y^2} s_2\left(x_i, y_j\right) = f_{ij}^{0,2}, & i = 0, \Lambda\ , n; j = 0, m \\ \dfrac{\partial}{\partial x^2 \partial y^2} s_2\left(x_i, y_j\right) = f_{ij}^{2,2}, & i = 0, n; j = 0, m \end{cases}$$

具体求解过程略。

双二次样条函数的另一种表示方法是采用 B 样条表示。

在四个角点处取作两个单方向的二次 B 样条三重节点，$\triangle_x$ 与 $\triangle_y$ 网格的中点为单节点，即

$$t_1 = t_2 = t_3 = 0,$$

$$t_{i+3} = x_{i-1/2} \quad i = 1, 2, \Lambda\ , n,$$

$$t_{n+6} = t_{n+5} = t_{n+4} = 1$$

$$r_1 = r_2 = r_3 = 0$$

$$r_{j+3} = y_{j-1/2} \quad j = 1, 2, \Lambda\ , m$$

$$r_{m+6} = r_{m+5} = r_{m+4} = 1$$

以上列式为节点可以分别将 x 方向和 y 方向的二次样条唯一表示为

$$s_{2,\Delta x}\left(x\right) = \sum_{i=1}^{n+3} \alpha_i B_{i,3}\left(x\right)$$

及

$$s_{2,\Delta y}\left(y\right) = \sum_{j=1}^{m+3} \beta_j B_{j,3}\left(y\right)$$

这里 $B_{i,3}(x)$ 与 $B_{j,3}(x)$ 分别是以 $(t_i, t_{i+1}, t_{i+2}, t_{i+3})$ 及 $(r_j,\ r_{j+1},\ r_{j+2},\ r_{j+3})$ 为节点的二次 B 样条函数。由此可得

$$\begin{aligned} s_{2,\Delta} f = s_{2,\Delta x}\left(x\right) s_{2,\Delta y}\left(y\right) f &= \sum_{j=1}^{m+3} \beta_j B_{j,3}\left(y\right) s_{2,\Delta x}\left(x\right) \\ &= \sum_{j=1}^{m+3} \sum_{i=1}^{n+3} \alpha_{ij} B_{i,3}\left(x\right) B_{j,3}\left(y\right) \end{aligned}$$

若此时节点满足 Schoenberg 条件，则 $s_{2,\triangle}f$ 存在且唯一。

3 双三次样条

双变量函数 $s_{3,\triangle}(x,y)$称为区域Ω上按剖分△的双三次样条函数，是指它满足以下两个条件。

1) 在每个子矩形

$$\Omega_{i,j}=\{(x,y):x\in[x_i,x_{i+1}],y\in[y_j,y_{j+1}];\quad i=0,1,\Lambda\ ,n-1,y=0,1,\Lambda\ ,m-1\}$$

内 $s_{3,\triangle}(x,y)$是 x,y 的双三次多项式。

2) $s_{3,\triangle}(x,y)\in C^{2,2}(\Omega)$，即 $s_{3,\triangle}(x,y)$在区域Ω上有直到四阶的混合偏导数连续。

与 $s_{3,\triangle}(x,y)$一样，$s_{3,\triangle}(x,y)$的自由参数也为$(n+3)(m+3)$，即等于 x 与 y 两个方向一维三次样条自由参数的乘积。上面有双关二次样条的叙述可以平行移植到双三次样条。同理双三次样条的 B 样条形式可以表示为

$$s_{3,\Delta}f=\sum_{j=1}^{m+3}\sum_{i=1}^{n+3}\alpha_{ij}B_{i,4}(x)B_{j,4}(y)$$

3.4.2 工具箱中关于张量积函数的函数

Matlab 中样条工具箱所提供样条函数都可具有任意个变元，像张量积函数就是非单变元的样条函数。这些多变元的样条函数同样以两种标准的方式表示，即 B 形式和 pp 形式，并且它们的构造可以完全使用前面几节所介绍的函数和命令。接下来简单介绍一下如何利用已介绍的函数和命令构造张量积样条函数。为方便起见，以二元为例。

假设 f 是 x 的函数，g 是 y 的函数，那么它们的张量积为

$h(x，y) = f(x) * g(y)$

是 x 和 y 的函数，是一个二元函数。

更一般地讲，以 $s=(s_1,s_2,...,s_{m+h})$和 $t=(t_1,t_2,...,t_{n+k})$为节点序列，以$(a_{ij}: i=1,2,\cdots,m; j=1,2,\cdots,n)$为相应的参数阵列，则可得到二元样条函数

$$f(x,y)=\sum_{i=1}^{m}\sum_{j=1}^{n}B(x\mid s_i,\Lambda\ ,s_{i+h})B(y\mid t_j,\Lambda\ t_{j+k})a_{ij}$$

它可通过命令

```
sp = spmak({s, t}, a)
```

构造。进一步，可以用 fnplt、fnval、fnder、fnrfn、fn2fm 等命令对样条 sp 进行绘图、计算、微分、积分和转换等操作。更详细的例子见 tspdem。

虽然 Matlab 的样条工具箱中大部分对单变元样条操作的函数和命令也能对多变元的样条进行操作，Matlab 还是提供了专门的对多变元的样条进行操作的函数和命令，见表

3.4.1(注：这一部分函数在 Matlab 5.2 版本中，已经列为陈旧函数，失去功能，它们的任务已由单变元的函数和命令来完成，其原理见 3.6.1，这里对这些函数进行解释，出于对老版本，如 5.1 版用户的考虑)。

表 3.4.1　工具箱中关于张量积样条函数的函数

函　数　名	函 数 功 能
tcsapi	构造 not-a-knot 边界条件下的二元三次样条函数
tcsaps	构造二元三次光滑样条函数
tppbrk	分解一个 pp 形式的张量积样条函数
tppmak	构造一个 pp 形式的张量积样条函数
tppval	评价一个 pp 形式的张量积样条函数
tsp2pp	将双变量的张量积函数从 B 形式转化为 pp 形式
tpap2	最小方差的二元张量积样条逼近
tspapi	双变元张量积样条的插值
tspbrk	分解 B 形式的二元张量积函数
tspmak	构造一个 B 形式的二元张量积函数
tspval	评价一个 B 形式的二元张量积函数

1　tcsapi

功能：构造 not-a-knot 边界条件下的二元三次样条函数。

格式：tpp=tcsapi(x, y, x)

说明：用单变元的 csapi 构造双变元的三次样条函数，对于所有的 i，j，使

$$z(i,j)=f\left(x(i),y(j)\right)$$

其运行过程如下：

```
[by,cy,ly,ky,d] = ppbrk(csapi(y,z));
[bx,cx,lx,kx] = ppbrk(csapi(x,reshape(cy,d,ly*ky).'));
tpp = tppmak(bx,by,reshape(cx,ly*ky,lx*kx).');
```

函数的输出结果可用 fnder 进行微分，用 tppmak 分解，用 tppval 评价。

举例：

```
x=1:5;
y=2:4
z=[15:17;5:7;1:3;56:57;23:25];
tpp=tcsapi(x,y,z);
```

2　tcsaps

功能：构造二元三次光滑样条函数。

格式：tpp=tcsaps(x, y, z, px, py)

说明：用单变元的 csaps 构造双变元的三次样条函数，对于所有的 i,j 使

$$z(i,j)=f\left(x(i),y(j)\right)$$

其中，$nx=length(x)$，$ny=length(y)$，$i=1,\Lambda\ ,nx$；　$j=1,\Lambda\ ,ny$。

在 x 方向使用光滑参数 px，在 y 方向使用光滑参数 py。其运行过程如下：

```
nx = length(x);
ny = length(y);
tpp = tppmak(x,y, reshape(ppbrk(csaps(x,
reshape(ppbrk(csaps(y,z,py),'coefs'),nx,...
4*(ny-1)).',px),'coefs'),4*(ny-1),4*(nx-1)).');
```

举例:
```
x=1:5;
y=2:4
z=[15:17;5:7;1:3;56:57;23:25];
tpp=tcsaps(x,y,z,0.5,0.5);
```

3 tpbrk

功能: 分解一个 pp 形式的张量积样条函数。

格式: [breaksx, breaksy, coefs, l, k] = tppbrk(tsp[, print])

说明: 分解一个 pp 形式的张量样条函数 tsp，返回指定的部分，不过返回值的顺序不能改变。式中 breaksx 指 x 方向的节点序列，breaksy 指 y 方向的节点序列，coefs 为张量样条函数 tsp 的参数阵列。

举例:
```
x=1:5;
y=2:4
z=[15:17;5:7;1:3;56:57;23:25];
tpp=tcsaps(x,y,z,0.5,0.5);
[knotsx,knotsy,coefs]=tppbrk(tpp);
```

4 tppmak

功能: 构造一个 pp 形式的张量积样条函数。

格式: tpp = tppmak(breaksx, breaksy, coefs)

说明: 构造一个 pp 形式的张量积函数，并将信息存放于阵列中 tpp 中。假定 coefs 是一个大小为[*lx***kx*,*ly***ky*]的阵列。其运行过程如下:

```
[lkx,lky] = size(coefs);
lx = length(breaksx) ? 1;
kx = fix(lkx/lx);
ly = length(breaksy) ? 1;
ky = fix(lky/ly);
if (kx<=0)|(lx*kx~=lkx)|(ky<=0)|(ly*ky~=lky)
error('The input is incompatible.');
end
tpp=[20 lkx lky coefs(:).'kx breaksx(:).'ky breaksy(:).'];
```

举例:
```
x = 1:4;
y = 1:4;
```

```
coefs = [1:3;1:3;1:3];
tpp = tppmak(x, y, coefs);
```

5　tppval

功能：评价一个 pp 形式的张量积样条函数。

格式：values = tppval(tpp, x)

values = tppval (tpp, x, y)

说明：第一种格式返回 values (i) = f(X(:, i))的值，其中 X(:, i)代表阵列的第 i 列。第二种格式返回 values (i, j) := f(X(i),Y(j))的值。

举例：
```
x=1:5;
y=2:4
z=[15:17;5:7;1:3;56:57;23:25];
tpp=tcsaps(x,y,z,0.5,0.5);
values=tppval(tpp,x,y);
```

6　tsp2pp

功能：将双变量的张量积函数从 B 形式转化为 pp 形式。

格式：tpp=tsp2pp(tsp)

说明：将双变量的张量积函数从 B 形式转化为 pp 形式，其运行过程如下
```
[knotsx,knotsy,coefs] = tspbrk(tsp);
[breaksy,coefy,ly,ky,nx] =
ppbrk(sp2pp(spmak(knotsy,coefs)));
lky=ly*ky;
[breaksx,coefs,lx,kx,lky] =
ppbrk(sp2pp(spmak(knotsx,reshape(coefy,nx,lky).')));
tpp=tppmak(breaksx,breaksy,reshape(coefs,lky,lx*kx).')
```
该过程的中的一些函数将在下面介绍。

举例：
```
kx = 4;
knotsx=augknt([0:.2:1],kx);
ky = 3;
knotsy = augknt([0,.25,.5,.75,1],ky);
tsp = tspap2(knotsx,kx,knotsy,ky,x,y,z);
tpp=tsp2pp(tsp);
```

7　tspap2

功能：最小方差的二元张量积样条逼近。

格式：ysp = tspap2(knotsx, kx, knotsy, ky, x, y, z)

说明：x 方向以 knotsx 为节点序列，以 kx 为阶；y 方向以 knotsy 为节点序列，以 ky 为阶，返回在最小方差的意义下最为符合(x(i), y(j), y(i, j)), i=1, ..., length(X),

j=1, ..., length(Y)的双变元张量积函数。

举例：
```
x = sort([[0:10]/10,.03 .07, .93 .97]);
y = sort([[0:6]/6,.03 .07, .93 .97]);
z = franke(x.'*ones(1,length(y)),ones(length(x),1)*y);
kx = 4;
knotsx=augknt([0:.2:1],kx);
ky = 3;
knotsy = augknt([0,.25,.5,.75,1],ky);
tsp = tspap2(knotsx,kx,knotsy,ky,x,y,z);
```

8 tspapi

功能：双变元张量积样条的插值。

格式：tsp = tspapi(knotsx, knotsy, x, y, z)

说明：x 方向以 knotsx 为节点序列，y 方向以 knotsy 为节点序列，返回最为符合

(x(i), y(j), z(i, j)), i=1, ..., length(x), j=1, ..., lengthx)

的双变元张量积函数。

所构造的函数 x 方向阶数为

length(knotsx) - length(x)

y 方向阶数为

length(knotsy) - length(y)

举例：
```
x = sort([[0:10]/10,.03 .07, .93 .97]);
y = sort([[0:6]/6,.03 .07, .93 .97]);
z = franke(x.'*ones(1,length(y)),ones(length(x),1)*y);
kx = 4;
knotsx=augknt([0:.2:1],kx);
ky = 3;
knotsy = augknt([0,.25,.5,.75,1],ky);
tsp = tspapi(knotsx,knotsy,x,y,z);
```

9 tspbrk

功能：分解 B 形式的二元张量积函数。

格式：[knotsx, knotsy, coefs, n, k] = tspbrk(tsp[, print])

说明：返回值中，knotsx 指 x 方向的节点序列，knotsy 指 y 方向的节点序列，coefs 为二元张量积函数 tsp 的参数阵列，n 和 k 均是包含两个元素的行向量，且

n=[length(x), length(y)]

k=[length(knotsx)-length(x), length(knotsy)-length(y)]

其中 x, y, knotsx, knotsy 的内容见下面举例。

举例：
```
x = sort([[0:10]/10,.03 .07, .93 .97]);
```

```
y = sort([[0:6]/6,.03 .07, .93 .97]);
z = franke(x.'*ones(1,length(y)),ones(length(x),1)*y);
kx = 4;
knotsx=augknt([0:.2:1],kx);
ky = 3;
knotsy = augknt([0,.25,.5,.75,1],ky);
tsp = tspapi(knotsx,knotsy,x,y,z);
[knotsx,knotsy,coefs,n, k]=tspbrk(tsp);
```

10　tspmak

功能：构造一个 B 形式的二元张量积函数。

格式：tsp = tpmak(knotsx, knotsy, coefs)

说明：式中 knotsx, knotsy 分别为 x, y 方向的节点序列，coefs 为构造二元张量积函数的参数阵列，且要求

length(knotsx)-r>0

length(knotsy)-c>0

其中[r, c]=size(coefs)。

举例：

```
knotsx = augknt([0:.2:1],kx);
knotsy = augknt([0,.25,.5,.75,1],ky);
coefs = [1:4; 1:4; 1:4; 1:4; 1:4];
tsp = tspmak(knotsx,knotsy,coefs);
```

11　tspval

功能：评价一个 B 形式的二元张量积函数。

格式：values=tspval(tsp, x, p)

说明：返回 values (i, j) := f(x(i), y(j))的值，其中二元张量积函数 f 的信息以 B 形式存放其中。

举例：

```
x = sort([[0:10]/10,.03 .07, .93 .97]);
y = sort([[0:6]/6,.03 .07, .93 .97]);
z = franke(x.'*ones(1,length(y)),ones(length(x),1)*y);
kx = 4;
knotsx=augknt([0:.2:1],kx);
ky = 3;
knotsy = augknt([0,.25,.5,.75,1],ky);
tsp = tspapi(knotsx,knotsy,x,y,z);
values = tspval(tsp,x,y);
```

3.5 其 他 函 数

Matlab 的样条工具箱中不仅包含以上的一些专用的具有针对性的函数，而且提供一些通用的函数。它们中的一些既可以对 pp 形式的样条函数进行操作，又可以对 B 形式的样条函数进行操作；另一些则提供额外的功能，例如对节点的操作。本节将分三类介绍这些函数：

- 对样条函数进行操作的函数
- 对节点进行操作的函数
- 独立函数

3.5.1 对样条函数进行操作的函数

样条工具箱提供了表 3.5.1 的对样条函数进行操作的函数。

表 3.5.1 对样条函数进行操作的函数

函 数 名	函 数 功 能
fnbrk	分解样条函数
fncmb	函数运算
fn2fm	将样条函数从一种形式转化为另一种形式
fnder	对样条函数进行微分
fnint	对样条函数进行积分
fnjmp	求样条函数在某一点的跳跃值
fnplt	绘图
fnval	评价一个样条函数

1 fnbrk

功能：分解样条函数。

格式：out = fnbrk(f, part)

pp = fnbrk (pp, [a b])

pp = fnbrk(pp, j)

fnbrk(f)

说明：第一种格式下该函数返回样条函数 f 中由 part 指定的部分，样条函数 f 既可以为 B 形式也可以为 pp 形式。当 part 为字符串‘form’时，函数返回样条函数 f 的形式，即函数 f 为 B 形式还是 pp 形式。当样条函数 f 为 B 形式时，part 可为以下字符串：‘knots’或‘t’、‘coefs’、‘number’、‘order’、‘dim’和‘interval’，它们的含义见 spbrk 的说明。当样条函数 f 为 pp 形式时，part 可为以下字符串：‘breaks’、‘coefs’、‘pieces’或‘l’、‘order’、‘dim’和‘interval’，它们的含义见 ppbrk 的说明。当函数 f 为多变元时，则返回相应变元的指定部分。第二、第三种格式仅适用于 pp 形式的样条函数，分别返回样条函数在区间[a b]内的部分和样条函数的第 j 片。第四种格式下，没有返

回值，但是将函数的各个部分显示在屏幕上。

举例：
```
[xx,yy]=titanium;
pick=[1 5 11 21  27 29 31 33 35 40 45 49];
tau=xx(pick);
y=yy(pick);
n=length(tau);
dl=tau(2)-tau(1);dr=tau(n)-tau(n-1);
t=[tau(1)-dl*[2 1] tau tau(n)+dr*[1 2]];
sp=spapi=spapi(t,tau,y);
out = fnbrk(sp,   rder? ;
```

2 fncmb

功能：函数运算。

格式：fn = fncmb(function, matrix)

fn = fncmb(function, function)

fn = fncmb(function, matrix, function)

fn = fncmb(function, matrix1, function, matrix2)

fn = fncmb(function, 'op', function)

说明：该函数提供一种简单的对样条函数进行线性缩放和组合的方法， 特别当样条函数用矢量或矩阵表示时，可以对样条函数进行简单的矩阵运算。第一种格式为对一个样条函数放大 matrix 倍；第二种格式为求两个相同样条函数的和；第三种格式是将第一个样条函数放大 matrix 倍后加到第二个样条函数函数上；第四种格式是分别将第一个样条函数和第二个样条函数函数放大 matrix1 倍和 matrix2 倍后加到一起；第五种格式中，op 可分别为‘+’、‘-’、 ‘*’，分别代表求两个样条函数的和、差、积。

举例：
```
fncmb(fn,3.5)
fncmb(f,3,g,-4)
fncmb(f,3,g)，其中 f,g 为样条函数。
```

3 fn2fm

功能：将样条函数从一种形式转化为另一种形式。

格式：g = fn2fm(f, form)

说明：将样条函数 f 转化为字符串参数 form 所指定的形式。form 可为以下字符串：

'pp'——转化为分段多项式形式，

'sp'或 'B-'——转化为 B 形式的样条函数，

'bb'或'BB'——转化为 BB 形式的样条函数。

BB 形式的样条函数是样条函数的一种特殊情况，它的节点序列中的每一个点均为最高重数 k。

举例：
```
sp = fn2fm(spline(x,y),'sp')
```

4 fnder

功能：对样条函数进行微分。

格式：fprime = fnder(f)

fprime = fnder(f, dorder)

说明：函数返回样条函数 f 的第 dorder 阶微分，dorder 的缺省值为 1。当 dorder 为负数时，函数返回以 dorder 的绝对值为阶的样条函数 f 的不定积分。输入样条函数为何种形式，则返回何种形式的样条函数。当输入样条函数为 m 变元时，dorder 必须明确给出，且其长度为 m。

举例：

```
x = sort([[0:10]/10,.03 .07, .93 .97]);
y = sort([[0:6]/6,.03 .07, .93 .97]);
z = franke(x.'*ones(1,length(y)),ones(length(x),1)*y);
kx = 4;
knotsx=augknt([0:.2:1],kx);
ky = 3;
knotsy = augknt([0,.25,.5,.75,1],ky);
tsp = tspap2(knotsx,kx,knotsy,ky,x,y,z);
fprime = fnder(tsp,[2,2])
```

5 fnint

功能：对样条函数进行积分。

格式：intgrf = fnint(f)

intgrf = fnint(f, value)

说明：fnint(f, value)是指对单变元的样条函数 f 的不定积分，并将积分正规化，使其在样条函数 f 的基本区间的左端点等于指定的值 value，缺省情况下为零；并且输出所得函数为与输入函数同样形式的样条函数。当对多变元的样条函数 f 进行不定积分时，可以使用 fnder(f, dorder)，其中 dorder 为一个非正的向量。

举例：

```
[xx,yy]=titanium;
pick=[1 5 11 21  27 29 31 33 35 40 45 49];
tau=xx(pick);
y=yy(pick);
n=length(tau);
dl=tau(2)-tau(1);dr=tau(n)-tau(n-1);
t=[tau(1)-dl*[2 1] tau tau(n)+dr*[1 2]];
sp=spapi(t,tau,y);
intgrf = fnint(sp);
```

6 fnjmp

功能：求样条函数在某一点的跳跃值。

格式：jumps = fnjmp(f, x)

说明：函数返回样条函数 f 在点 x 处的右极限值与左极限值的差，即 f(x+)−f(x−) 的值，并且 x 不仅仅可以为独立的点，而且可以为一个点的序列。

举例：
```
pp = ppmak(1:4,1:3),
fnjmp(pp,1:4);
x = cos([4:-1:0]*pi/4),
sp = spmak ((x,1),3);
fnjmp(fndersp,x);
```

7　fnplt

功能：绘图。

格式：fnplt(f)

fnplt(f, arg1, arg2, arg3)

points = fnplt(f)

说明：函数 fnplt 绘出样条函数 f 的图形，arg1、arg2、arg3 三个参数分别为线宽 linewidth、绘图区间 interv 和样式 symbol。线宽 linewidth 是一个数值型变量，interv 指定一个绘图区间，样式 symbol 可以指定图形的颜色、绘图方式等，其具体内容参见函数 plot 的函数说明。值得说明的是这三个参数的出现次序为任意的。当使用该函数并明确给出返回值 points 时，则函数不画出任何图形，而是计算出 f(x)的值存放于 points 中。

举例：
```
pp = ppmak(1:4,1:3),
fnplt(pp);
fnplt(pp,   ? [1 4],3);
points = fnplt(pp)
```

8　fnval

功能：评价一个样条函数。

格式：values = fnval(f, x)

说明：fnval(f, x)返回 f 所代表的样条函数在参数 x 所规定的点的函数值，记为 f(x)。当样条函数 f 为单变元、输入为参数 x 的大小为[m, n] 并且 f 的目标为 d 维时，得到的输出矩阵的大小为[d*m, n]。如果样条函数 f 在某一点是不连续的，则 fnval 返回样条函数 f 在该点的右极限值 f(x+)，特殊情况下，即该点为右端点时，返回左极限值 f(x−)。当样条函数为 m（m>1）变元时，输入参数 x 必须包含 m 个向量，例如为一个大小为[m，n]的矩阵或每个元素均为向量的队列{x1, ..., xm}，在第一种情况下 fnval 返回样条函数 f 在 x 上的值，且为一个大小为[d*m，　n]的矩阵；第二种情况下返回值的大小为[d, length(x1),..., length(xm)]。

举例：
```
pp = ppmak(1:4,1:3),
```

```
values = fnval(pp,1:4)
```

3.5.2 对节点进行操作的函数

样条工具箱提供表 3.5.2 的对节点进行操作的函数。

表 3.5.2 对节点进行操作的函数

函 数 名	函 数 功 能
augknt	扩张节点序列
aveknt	提供节点的平均值
brk2knt	按指定的重数由一个插值点序列产生一贯节点序列
knt2brk	将节点序列转化为插值点序列
knt2mlt	返回节点序列重数
newknt	改进插值点序列的分布
optknt	为插值提供优化的节点序列
sorted	关于某一节点序列将另一节点序列定位其中

1 augknt

功能：扩张节点序列。

格式：[augknot, addl] = augknt(knots, k)

[augknot, addl] = augknt(knots, k, mults)

说明：该函数返回一个不减的扩张的节点序列，使第一个和最后一个节点为 k 重，并且可选的返回节点序列左边增加的节点个数 addl，如果 k 为负值，实际上可能缩短节点序列的长度，并且使 addl 的值为负值。如果给出参数 mults，且为一个数量，那么待扩张的节点序列的每一个内节点，即不是端点的节点，都将重复 mults 次。如果 mults 为一个向量，且它的长度等于内节点的个数，那么第 j 个内节点将为 mults(j)重；如果不等，则所有的内节点将为 mults(1)重，且缺省值为 1。如果 knots 的内节点是严格增长的，这就使以 augknot 为节点序列的 k 阶样条函数在第 j 个内节点满足 k-mults(j)阶的光滑条件。

举例：

```
x=1:4;
augknot = augknt(x,3);
augknot =
    1    1    1     2     3     4     4     4

augknot = augknt(x,3,2:3);
augknot =
    1    1    1    2    2    3    3    3    4     4     4
```

2 aveknt

功能：提供节点的平均值。

格式：tstar = aveknt(t, k)

说明：该函数返回连续的 k-1 个节点的平均值，即

$$tstar(i)=\frac{t(i)+t(i+1)+\Lambda\ +t(i+k-1)}{k-1}$$

且 tstar 序列可以作为一个比较好的插值点序列。

举例：
```
t = augknt(breaks,k);
x = aveknt(t);
sp = spapi( t , x, sin(x) );
```

3 brk2knt

功能：按指定的重数由一个插值点序列产生一贯节点序列。

格式：knots = brk2knt(breaks, mults)

[knots, index] = brk2knt(breaks, mults)

说明：该函数返回的 knots 序列是由 breaks 序列中的第 i 个元素出现 mults(i)次构成的，i=1: length(breaks)。在特殊情况下，即 mults(i)<0 时，breaks(i)将不会在 knots 中出现。如果 mults 的单元数不等于序列 breaks 的长度，那么令所有的 mults(i)= mults(1)。如果 breaks 序列是严格增长的，并且所有的 mults(i)均大于零，则 index(i)为 breaks(i)在 knots 序列中第一次出现的位置。

举例：
```
x = 1:3;
knots = brk2knt(x,3:5);
knots =
1    1    1    2    2    2    2    3    3    3    3    3
```

4 knt2brk

功能：将节点序列转化为插值点序列。

格式：[breaks, mults] = knt2brk(knots)

说明：该函数功能与 brk2knt 相反，将节点序列 knots 中重复的节点去掉，然后将结果以严格递增次序存放于序列 breaks 中，即序列 breaks 中的任何点均为一重。mults(i)为 breaks(i)在节点序列 knots 中出现的次数，即重数。

举例：
```
knots = [0 0 1 1 1 5 5 6 6 6 9 9];
[breaks,mults] = knt2brk(knots);
```
返回值为
```
breaks = [0 1 5 6 9];
mults = [2 3 2 3 2]。
```

5 knt2mlt

功能：返回节点序列重数。

格式：[m, t] = knt2mlt(t)

说明：该函数返回节点序列 t 的重数 m 和排序后的节点序列 t。

举例：
```
x = [ 1 2 3 3 1 3];
[m,t] = knt2mlt(x);
```
返回值为
```
m = [0 1 0 0 1 2];
t = [1 1 2 3 3 3]。
```

6 newknt

功能：改进插值点序列的分布。

格式：newbreaks = newknt(pp, newl)

[newbreaks, distfn] = newknt(pp, newl)

说明：该函数返回一个更加合理分布的插值点序列，它在 pp 所表示的样条函数的基本区间上将样条函数重新划分为 newl 段，并给出新的插值点序列 newbreaks，在第二种格式下还给出新的 pp 形式的线性单调的样条函数 distfn，且要求 distfn 的高阶导数是等分布的，并且前面的划分也是在这样一种意义下进行的。该函数的意图是得到一个更好的插值点序列来更好地逼近 pp 所表示的样条函数，pp 是一个大致的逼近，且认为它包含足够的信息。

举例：
```
t = linspace(X(1),X(end),M+1);
knots = augknt(t,K);
sp = spap2(knots,K,X,Y);
breaks = newknt(sp,M);
sp = spap2(augknt(breaks,K),K,X,Y)。
```

7 optknt

功能：为插值提供优化的节点序列。

格式：knots = optknt(tau, k)

说明：该函数按照 Micchelli/Rivlin/Winograd 和 Gaffney/Powell 优化恢复理论返回由节点序列 tau 所确定的 k 阶样条函数 $s_{k,tau}$ 的最佳差值点序列 t。

举例：
```
sp = spapi(augknt([x([1 end]) optknt(x,k)],k),x,y);
```

8 sorted

功能：关于某一节点序列将另一节点序列定位其中。

格式：pointer = sorted(meshpoints, points)

说明：该函数首先将 meshpoints 和 points 两个节点序列按不减顺序排序，然后将序列 points 中的第 i 个节点 points(i)插入到序列 meshpoints 中的第 j 个节点 meshpoints(j) 的后面，满足条件 meshpoints(index(j)) ≤ points(j) < meshpoints(index(j)+1)，且得到 pointer(i) = j，其中 i = 1: length(points)。

举例：
```
x = [8 4 6 3];
y = [3 5 9 1];
```

```
pointer = sorted(x, y);
```
返回值为
```
pointer = [0 1 2 4]。
```

3.5.3　独立函数

样条工具箱提供了以下独立函数(见表 3.5.3)。

表 3.5.3　独立函数

函　数　名	函 数 功 能
slvblk	求解一个类块对角矩阵的线性系统
bkbrk	返回块对角矩阵的细节
franke	Franke 的双变元测试函数
subplus	取正函数
titanium	测试数据

1　slvblk

功能：求解一个类块对角矩阵的线性系统。

格式：x = slvblk(blokmat, b)

x = slvblk(blokmat, b, w)

说明：slvblk(blokmat, b)返回线性系统 Ax = b 的解，A 存放于 blokmat 中。如果方程组是超定的，例如方程个数大于未知数的个数，则返回最小方差意义下的解，在这种情况下，可以给出可选参数 w 确保解 x，使

sum_j W(j)*((A*X-B)(j))^2

为最小。

举例：
```
sp=spmak(knots,slvblk(spcol(knots,k,x,1),y.'))
```

2　bkbrk

功能：返回类块对角矩阵的细节。

格式：[nb, rows, ncols, last, blocks] = bkbrk(blokmat)

bkbrk(blokmat)

说明：该函数是 slvblk 中使用的一个工具，返回包含在 blokmat 中的类块对角矩阵的细节信息。

举例：
```
bkbrk(spcol([1:6],3,.1+[2:4],'sl'));
```
返回值为
```
block 1 has 2 row(s)
     0.5900    0.0050         0
     0.4050    0.5900    0.0050

next block is shifted over 1 column(s)
block 2 has 1 row(s)
```

```
        0.4050    0.5900
    next block is shifted over 2 column(s)
```

3 franke

功能：Franke 的双变元测试函数。

格式：values = franke(x, y)

说明：返回 Franke 测试函数 f(x, y)的数据。其原理如下：

```
values = .75*exp(−((9*x−2).^2 + (9*y−2).^2)/4) + ...
        .75*exp(− ((9*x+1).^2)/49 − (9*y+1)/10) + ...
        .5*exp(− ((9*x−7).^2 + (9*y−3).^2)/4) - ...
        .2*exp(− (9*x−4).^2 - (9*y−7).^2);
```

举例：`values = franke(x,y)`

4 subplus

功能：取正函数。

格式：y = subplus(x)

说明：y = subplus(x) := (x)_{+} $=\begin{cases} x, x \ge 0 \\ 0, x < 0 \end{cases}$

5 titanium

功能：测试数据。

格式：[x, y]=titanium

说明：返回测试数据，所得如下：

```
x=585+[1:49]*10;
y=[.644  .622  .638  .649  .652  .639  .646  .657  .652  .655  ...
   .644  .663  .663  .668  .676  .676  .686  .679  .678  .683  ...
   .694  .699  .710  .730  .763  .812  .907 1.044 1.336 1.881];
y=[ y ...
   2.169 2.075 1.598 1.211  .916  .746  .672  .627  .615  .607  ...
    .606  .609  .603  .601  .603  .601  .611  .601  .608];
```

举例：

```
x = titanium;
[x,y] = titanium;
```

3.6　举　　例

为了更好地了解 Matlab 中样条工具箱的原理和应用，本节给出两个具体的例子，供读者参考。

3.6.1　使用张量积样条函数对多变元函数的近似法

因为样条工具箱能够处理具有向量参数的样条函数，所以它能容易地通过张量积样条函数对网格点数据进行插值和逼近。下面例子中将详细地解释，同时读者可以参见工具箱提供的演示程序 tspdem。

必须明确，大部分的张量积样条函数对网格点数据的逼近可以直接由工具箱提供的样条函数的创建命令得到(比如 spapi 和 csape 命令)，而不用关心在本例子中所讨论的具体细节。更确切地说，这个例子的含义是解释创建张量积样条函数背后的理论，这些内容将有助于理解工具箱中创建命令没有覆盖到的部分。

考虑对给定数据 z(i, j) = f(x(i), y(j)), i = 1、…、Nx, j = 1、…、Ny 的最小方差逼近。我们通过工具箱中的 Franke 命令来得到数据。为了更好地解释，我们在 x 方向和 y 方向选择更多的数据点，并在端点附近使数据点更加密集。令

```
x = sort([[0:10]/10,.03 .07, .93 .97]);
y = sort([[0:6]/6,.03 .07, .93 .97]);
```

得到

```
[yy,xx] = meshgrid(y,x);
z = franke(xx,yy);
```

认为这些数据来自以向量为值的函数，即对于所有的 j, y 的函数在 y(j)处的值为向量 z(:,j)。没有特殊的原因，我们选择抛物线样条函数来逼近函数 z，并选择三个均匀分布的内节点。这样就意味着确定了样条函数的阶数和节点序列：

```
ky = 3;
knotsy = augknt([0,.25,.5,.75,1],ky);
```

然后利用命令 spap2 来构造在最小方差意义下逼近以上数据的样条：

```
sp = spap2(knotsy,ky,y,z);
```

实际上，我们通过 $S_{ky,knotsy}$ 同时逼近了 x 方向上离散的 Nx 个数据，即

$$\left(y(j), z(i,j)\right)_{j=1}^{Ny} \qquad i = 1, \Lambda\ , Nx$$

特别对于以下命令

```
yy = [-.1:.05:1.1];
vals = fnval(sp,yy);
```

返回的阵列 vals，它的列 vals(:, j)是样条曲线 sp 在点 yy(j)处的值，它的元素 vals(i, j)相当于对函数 f 在网格点(x(i), yy(j))的函数值 f(x(i), yy(j))的近似。这在用如下命令得到的图形（图

3.6.1）中非常明显：

```
mesh(x,yy,vals.');
view(150,50)
```

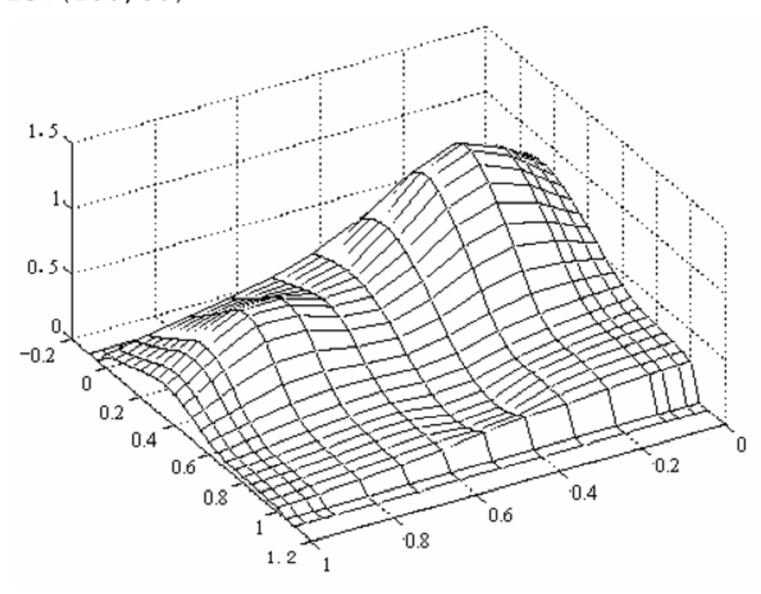


图 3.6.1　一簇光滑曲线拼成的表面

注意，命令 mesh 中用的是 vals.'，为 vals 的转置，这是由于 Matlab 自身作图视角的需要。在前面得到的 z 和 vals 是按标准的逼近理论排列的，即不同的行代表不同的 x，而 mesh 和 surf 等命令却不同，它们以不同的行代表不同的 y。这一点一定要注意，有可能引起严重的错误。

注意到图 3.6.1 中，任意一条光滑曲线的起始两点和最后两点的实际值为零，这是因为节点序列 yy 的起始两点和最后两点在样条 sp 的基本区间之外。再注意到曲线的隆脊，使我们确信，绘出的曲线只在一个方向上是光滑的。

为了得到实际的表面，我们必须更深入一步。考虑 sp 所表示的样条函数的参数阵列 coefsy

```
coefsy = fnbrk(sp,'c');
```

理论上可将样条函数 sp 看成函数

$$y \to \sum_r coefsy(r,:)B_{r,ky}(y)$$

其中参数行向量 coefsy(r, :)的第 i 个元素 coefsy(r, i)对应于 x(i)(对于所有的 i)。这就意味着对于参数阵列 coefsy 的每一个行向量 coefsy(r, :)有一个同样阶数 kx 和同样节点序列 knotsx 的样条函数在逼近。同样没有任何特殊的原因，这次我们选择有四个均匀分布内节点的三次样条函数，如下列命令：

```
kx = 4;
knotsx = augknt([0:.2:1],kx);
sp2 = spap2(knotsx,kx,x,coefsy.');
```

注意到 spap2(knots,k,x,fx)期望 fx(:, j)是 x(j)处的数据，即期望 fx 的每一列是一个函数的值。因为需要让 coefsy(r, :) 和 x(r)配合,所以必须将 coefsy 进行转置。

现在考虑作为结果的样条 sp2 的转置后的参数阵列 coefs 为

```
coefs = fnbrk(sp2,'c').';
```

它提供了二元样条函数

$$(x,y)\rightarrow\sum_{q}\sum_{r}coefs(q,r)B_{q,kx}(x)B_{r,ky}(y)$$

对初始数据

$$(x(i),y(j))\rightarrow z(x(i),y(j)),\quad i=1，\cdots，Nx\ ；\quad j=1\ ，\cdots，\ Ny$$

的逼近。为了绘出样条函数的图形，取网格为

```
xv = [0:.025:1];
yv = [0:.025:1];
```

执行以下命令

```
values = spcol(knotsx,kx,xv)*coefs*spcol(knotsy,ky,yv).';
mesh(xv,yv,values.');
view(150,50);
```

得到图形 3.6.2。

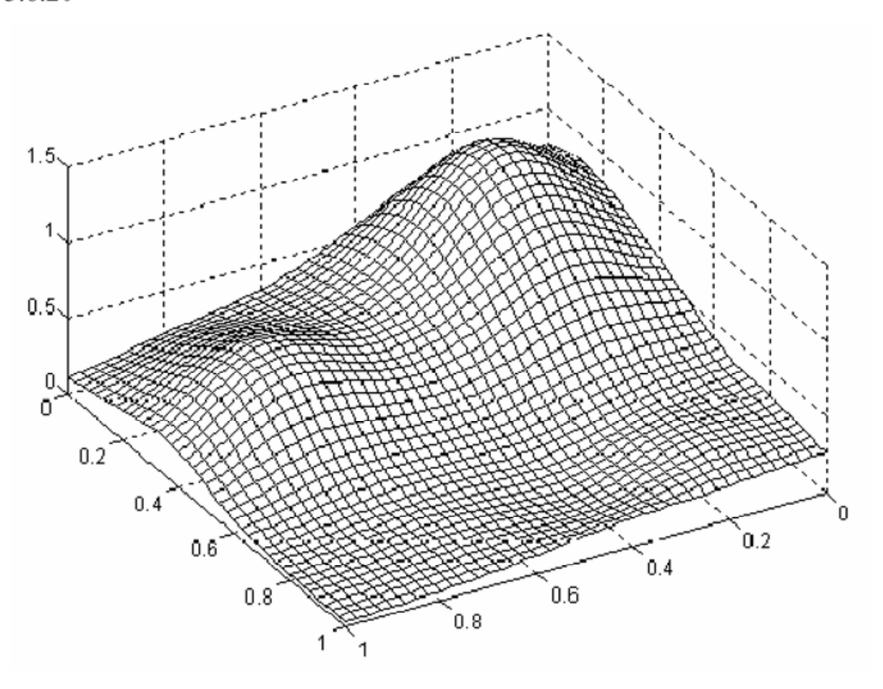


图 3.6.2　样条函数对 Franke’s 函数的逼近

这非常有意义，因为矩阵 spcol(knotsx, kx, xv)的第(i, q)个元素等于第 q 个以 knotsx 为节点序列的 kx 阶 B 样条函数 $B_{q,kx}(xv(i))$在 $xv(i)$处的值。既然矩阵 spcol(knotsx, kx, xv) 和矩阵 spcol(knotsy, ky, yv)被结合在一起，那么利用 fnval 计算也许会更加有效(虽然可能花费更多的内存)：

```
value2 = fnval(spmak(knotsx,...
fnval(spmak(knotsy,coefs),yv).'),xv).';
```

事实上这就fnval直接被一个张量积样条函数所调用时，即

```
value2 = fnval(spmak({knotsx,knotsy},coefs),{xv,yv});
```

内部所发生的。

接下来计算相对误差，也就是给定数据和由近似法所得的值之间的差别，可以按以下方式求得：

```
errors = z - spcol(knotsx,kx,x)*coefs*spcol(knotsy,ky,y).';
disp( max(max(abs(errors)))/max(max(abs(z))) )
0.0539
```

这也许不能令人信服。另一方面，我们用一个大小为

```
disp(size(coefs))
8 6
```

的参数阵列来拟合大小为

```
disp(size(z))
15 11
```

的数据阵列。这里使用的方法看起来是不太合理的：因为我们首先考虑z是以向量为函数值的变量y的函数，然后用得到的近似曲线的参数矩阵作为以向量为函数值的变量x函数。如果反过来会发生什么呢，也就是说先考虑x后考虑y发生什么？答案是令人惊奇的，最终结果大致不变。下面是一个数值实验。

首先，我们用样条曲线来拟合数据，但是这一次以x为独立变量，这样z的行成为数据点。相应地必须将z转置后才能用于spap2，即

```
spb = spap2(knotsx,kx,x,z.');
```

得到对所有曲线(x; z(:, j))近似的样条spb。通过命令

```
valsb = fnval(spb,xv).';
```

矩阵valsb，它的元素valsb(i, j)可被视为f(xv(i), y(j))的近似值。执行如下命令

```
mesh(xv,y,valsb.');
view(150,50)
```

得到图形3.6.3。

注意到曲线的隆脊，再次使我们确信，绘出的曲线只在一个方向上是光滑的，只不过这次换了一个方向。

为了得到实际的表面，我们更进一步。首先用如下命令分解出样条spb的系数阵列

```
coefsx = fnbrk(spb,'c');
```

然后对该系数阵列的每一个行向量coefsx(r, :)用同样阶数ky和同样节点序列knotsy的样条函数去拟合,得

```
spb2 = spap2(knotsy,ky,y,coefsx.');
```

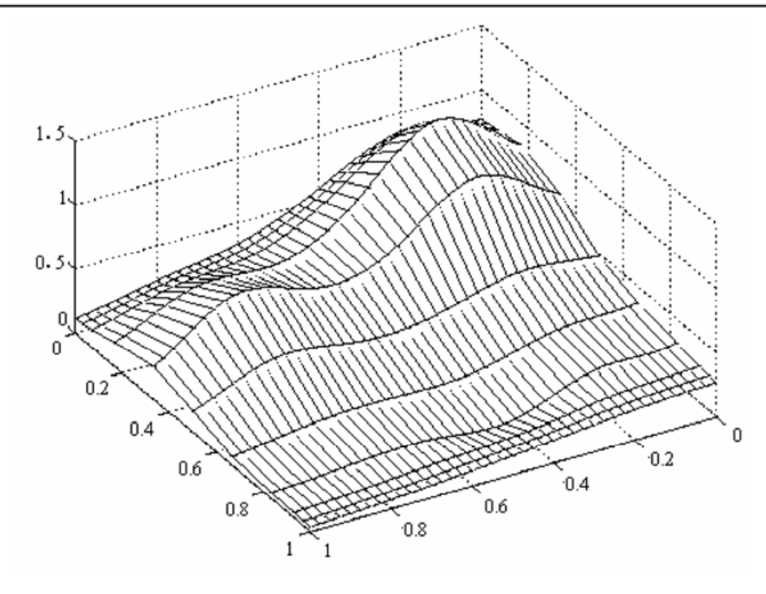


图 3.6.3　另一簇光滑曲线拼成的表面

同样，这里需要将coefsx进行转置。相对地，不需要将作为结果的曲线的参数阵列coefsb进行转置，得

```
coefsb = fnbrk(spb2,'c');
```

将其与前面得到的coefs比较：

```
disp( max(max(abs(coefs - coefsb))) )
4433e-5
```

由此可以看出二者是大致相等的。证明如下：

包含在sp = spap2(knots, k, x, y)中的样条函数s的系数阵列c线性的依靠纵坐标y。这就意味着存在这样一些矩阵，使

$$A = A_{knots,\,k,\,x}\,,$$

$$c = y * A_{knots,k,x}\text{，对于所有的数据 } y$$

这个陈述在y是一个矩阵时仍然成立。假设size(y) = [d, n]，在这种情况下，y的任何一列数据y(:, j)可被看为d 维空间的一个点，并且作为结果的样条函数的值是d维向量，且系数阵列 c 的大小为[d, n]，其中n = length(knots)−k。

特别在双变元时，应用第一种计算方法，首先计算y方向。下列命令

```
sp = spap2(knotsy,ky,y,z);
coefsy =fnpbrk(sp,'c');
```

为我们提供了系数阵列coefsy，且使之满足

$$\text{coefsy} \;= \text{z} * A_{knotsy,ky,y}$$

接下来计算

```
sp2 = spap2(knotsx,kx,xx,coefsy.');
```

```
coefs = fnbrk(sp2,'c').';
```

产生系数阵列coefs, 且使之满足

$$\text{coefs} = ((z * A_{knotsy,ky,y})' * A_{knotsx,kx,x})' = (A_{knotsx,kx,x})' * z * A_{knotsy,ky,y}$$

再应用第二种计算方法，首先计算x方向。下列命令

```
spb = spap2(knotsx,kx,x,z.');
coefsx = fnbrk(spb,'c');
```

为我们提供了系数阵列coefsy，且使之满足

$$\text{coefsx} = z' * A_{knotsx,kx,x}$$

接下来计算

```
spb2 = spap2(knotsy, ky, y, coefsx.');
coefsb = fnbrk(spb,'c');
```

得到

$$\text{coefsb} = \text{coefsx.}' * A_{knotsy,ky,y} = (A_{knotsx,kx,x}).' * z * A_{knotsy,ky,y}$$

这样就可以得到coefs = coefsb。证明完毕。

由上面的证明过程可知第二种方法比第一种方法更具对称性。因为在第二种方法中每次调用spap2都需要进行转置，并且其他地方不再需要，而且这种方法可以推广到任意个数变元的网格点数据。比如给定一个大小为[Nx, Ny, Nz]的三维数据点v，且

```
v(i,j,k) = f(x(i),y(j),z(k))
```

那么我们应从

```
coefs = reshape(v,Nx,Ny*Nz)
```

开始。假定nj = knotsj−kj, 对于 j = x, y, z，按如下过程进行：

```
sp = spap2(knotsx,kx,x,coefs.');
coefs = reshape(fnbrk(sp,'c'),Ny,Nz nx);
sp = spap2(knotsy,ky,y,coefs.');
coefs = reshape(fnbrk(sp,'c'),Nz,nx ny);
sp = spap2(knotsz,kz,z,coefs.');
coefs = reshape(fnbrk(sp,'c'),nx,ny nz);
```

以上方法就是当函数 csapi、sape、spapi、spaps、spap2 拟合网格点数据时使用的方法，同样也是 fnval 在评价张量积样条时使用的方法。

3.6.2 Chebyshev 样条函数的构造

Chebyshev 样条函数 $C = C_t = C_{k,t}$ 是以 t=(t_i:i=1,...,n+k)为节点序列的 k 阶样条函数，它

的最大范数为 1，并且在区间$[t_k,t_{n+1}]$内振荡，且其函数值在 t_{n+1} 附近为正。这就意味着存在一个严格递增的 n 元序列 τ，使由 $C(\tau_i)=(-1)^{n-1}$ 给定的函数 $C=C_t \in S_{k,t}$ 在区间$[t_k,t_{n+1}]$上最大范数为 1。由此可以知道 $\tau_1=t_k$，$\tau_n=t_{n+1}$ 和 $t_i<\tau_i<t_{k+i}$。实际上对于所有 i 存在 $t_{i+1}<\tau_i<t_{k+i-1}$。简而言之，Chebyshev 样条函数 C 与 Chebyshev 多项式相似，履行同样的功能。在这个例子中，我们通过一个特定的节点序列 t 来构造 Chebyshev 样条 C。

由于使用三次样条，则阶数

k = 4

插值点序列如下：

```
breaks = [0 1 1.1 3 5 5.5 7 7.1 7.2 8];
lp1 = length(breaks);
```

由此可以构造节点序列 t 为

```
t = breaks([ones(1,k) 2:(lp1-1) lp1*ones(1,k)]);
```

它还可以用以下命令得到：

```
t = augknt(breaks,k);
```

则自由度为

```
n = length(t) ? k;
```

使用节点的均值

$$\tau_i=\left(t_{i+1}+\Lambda+t_{i+k+1}\right)/\left(k-1\right)$$

为对序列 τ 的初始估计，并将序列 τ 作为较好的插值点使用。其可以由命令 aveknt 得到：

```
tau = aveknt(t,k);
```

画出对 C 第一次近似的样条函数 c 的图形(图 3.6.4)，c 对所有的 i 满足 $c(\tau_i)=(-1)^{n-1}$

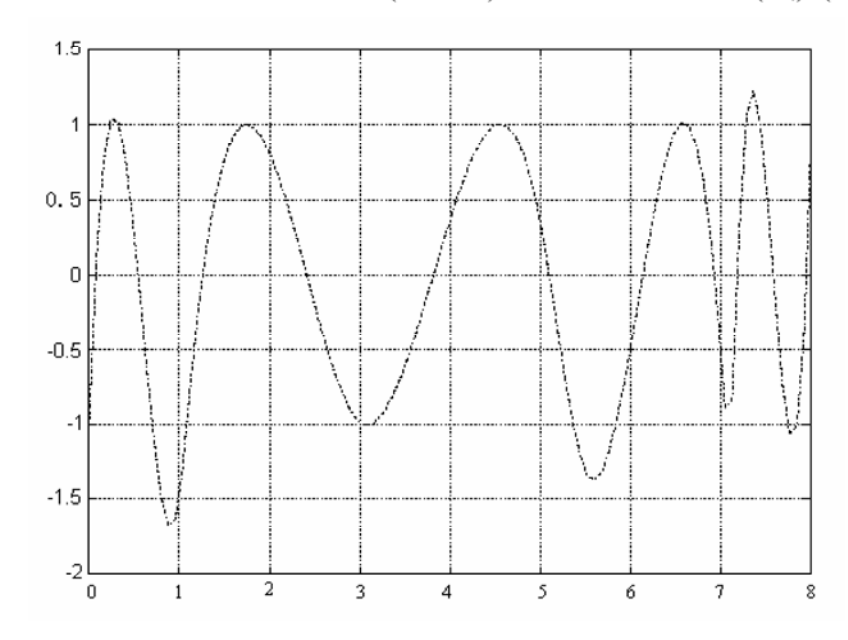


图 3.6.4　对 Chebyshev 样条的第一次近似

```
b = (-ones(1,n)).^[n?:?:0];
```

```
c = spapi(t,tau,b);
fnplt(c,'?')
grid
```

从这次近似开始，我们使用 Remez 算法产生一系列的样条向 C_t 会聚。这意味着我们将以当前近似的 c 的极值来构造新的序列 τ，然后再次向 C 逼近。

我们使用 c 的一阶微分 Dc 的零点作为新的 τ_i。首先对 c 进行微分

```
cp = fnder(c);
```

然后取 Dc 的控制多边形的零点作为对 Dc 零点的第一次推测。为了达到以上目的，必须分解样条 cp：

```
[knots, coefs, np, kp] = spbrk(cp);
```

控制多边形有极点(tstar(i), coefs(i))，其中 tstar 是样条函数的节点序列的平均值，表示为

```
tstar=aveknt(knots,kp);
```

下面就是样条 cp 的控制多边形的零点：

```
npp=[1:np-1];
guess=tstar(npp) - coefs(npp).*...
                  (diff(tstar)./diff(coefs));
```

这样就提供了对实际零点的一组很好的推测点。

将 tau 和上面的推测作为我们的第一次近似，对估计的 Dc 的零点通过两步正割法进行改进。首先，在两组点集上对 Dc 进行评价：

```
points = tau(ones(4,1),2:n-1);
points(1,:) = guess;
values = zeros(4,n-2);
values(1:2,:) = reshape(fnval(cp,points(1:2,:)),2,n-1);
```

接下来使用两步正割法。为了防止函数被零除，我们令函数值不同于 1。因为 Dc 在 points 附近是严格递增的，所以这是无害的。

```
for j=2:3
   rows=[j,j-1];
   cpd=diff(values(rows,:));
   cpd(find(cpd==0)) = 1;
   points(j+1,:) = points(j,:)...
       - values(j,:).*(diff(points(rows,:))./cpd);
   values(j+1,:) = fnval(cp,points(j+1,:));
 end
```

再执行命令

```
max(abs(values.'))
```

得到

```
ans =
      4.1176 5.7789 0.4644 0.1178
```

这个答案表明结果已经有所改进。现在我们用这些新的点作为构造新的节点 tau

```
tau = [tau(1) points(4,:) tau(n)];
```

然后计算极值得

```
extremes = abs(fnval(c, tau));
```

且 extremes 中最大元素和最小元素之间的差别：

```
max(extremes)-min(extremes)
ans = 0.6905
```

是我们离总水平有多远的估计。

按照新选择的节点序列 tau 构造一个新的样条函数并且将所得样条函数的图形画在旧图上得图 3.6.5。

```
c = spapi(t,tau,b);
points = sort([tau [0:100]*(t(n+1)-t(k))/100]);
values = fnval(c,points);
hold on,
plot(points,values)
```

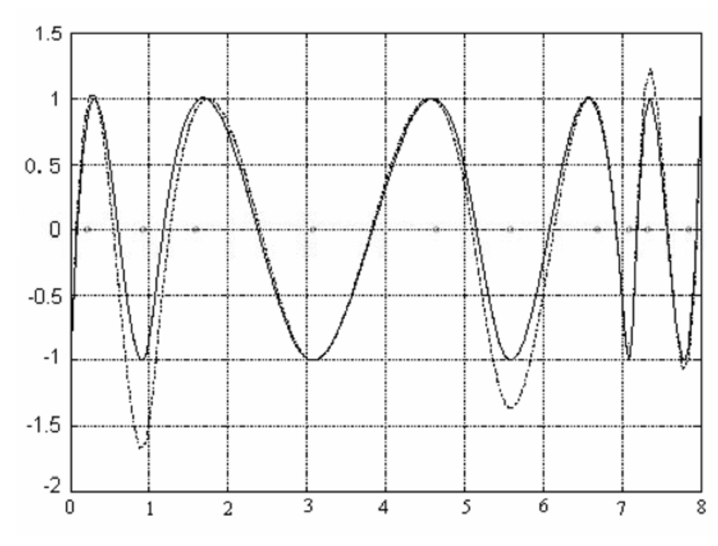


图3.6.5　一个更接近水平的样条

如果仍不够接近，那么只需继续这个循环即可。就这个例子来说，下一步迭代就可以得到精确的结果。

参 考 文 献

[1]　Spline Toolbox User's Guide，MathWorks, 1997

[2]　施法中著，计算机辅助几何设计与非均匀有理B样条，北京航空航天大学出版社，1991

[3]　方逵著，曲线曲面设计与显示原理，国防科技大学出版社，1993

[4]　孙家昶著，样条函数与计算几何，科学出版社，1990

第4章 优化工具箱

(Optimization Toolbox Ver 5.0)

优化理论是一门实践性很强的学科。它被广泛应用于生产管理、军事指挥和科学试验等各种领域，如工程设计中的最优设计，军事指挥中的最优火力配置问题等。优化理论和方法奠基于 20 世纪 50 年代。在二战期间，由于军事上的需要，提出并解决了大量的优化问题。但作为一门新兴学科，则是在 G.B.Dantzig 提出求解线性规划问题的单纯形法(1947)，H.W.Kuhnh 和 A.W.Tucker 提出非线性规划基本定理(1951)以及 R.Bellman 提出动态规划的最优化原理(1951)以后。之后，借助于计算机的发展，优化理论得到了飞速的发展，至今已形成具有多分支的综合学科。其主要分支有：线性规划、非线性规划、动态规划、图论与网络、对策论、决策论等。

4.1 优化工具箱简介

Matlab 的优化工具箱提供了对各种优化问题的一个完整的解决方案。其内容涵盖线性规划，二次规划、非线性规划、最小二乘问题、非线性方程求解、多目标决策、最小最大问题、以及半无限问题等的优化问题。其简洁的函数表达、多种优化算法的任意选择、对算法参数的自由设置，可使用户方便灵活地使用优化函数。

简单地，可以将优化工具箱中的函数分为以下 4 类，见表 4.1.1～表 4.1.4。

1 求极小值

表 4.1.1 优化函数列表

函 数 名	功 能
attgoal	求解多目标优化问题
constr	求解约束非线性优化问题
fmin	求解标量非线性优化问题
fminu,fmins	求解无约束非线性优化问题
lp	求解线性规划问题
minmax	求解最小最大问题
qp	求解二次规划问题
seminf	求解半无限问题

2 解方程

表 4.1.2 方程求解函数列表

函 数 名	功 能
\	线性方程求解
fsolve	非线性方程求解
fzero	标量非线性方程求解

3　最小二乘问题

表 4.1.3　最小二乘函数列表

函　数　名	功　　能
Conls	求解线性约束最小二乘最优问题
Curvefit	非线性数据拟合
leastsq	求解非线性最小二乘最优问题
nnls	求解非负约束最小二乘最优问题

4　优化函数控制

表 4.1.4　优化控制函数列表

函　数　名	功　　能
foptions	设置优化参数

4.2　优化工具箱基础

4.2.1　一个简单的例子

例 1　考虑如下优化问题

目标函数

$$\min_{x} f(x)=e^{x_1}\left(4x_1^2+2x_2^2+4x_1x_2+2x_2+1\right)$$

约束方程

$$x_1x_2-x_1-x_2\le -1.5$$
$$x_1x_2\ge -10$$

为了求解该优化问题，必须首先编写一个能够返回函数值的 M 文件，将函数表达式写入，然后调用有约束非线性优化函数 constr，由于优化函数要求约束方程具有 $G(x)\leqslant 0$ 的形式，因此必须将约束方程规范化，进行预处理。

规范化后约束方程变为

$$x_1x_2-x_1-x_2+1.5\le 0$$
$$-x_1x_2-10\le 0$$

求解过程

第一步：为目标函数即约束方程编写 M 文件——fun.m

```
function [f,g] = fun(x)
f = exp(x(1))*(4*x(1)^2+2*x(2)^2+4*x(1)*x(2)+2*x(2)+1);
g(1,1) = 1.5 + x(1)*x(2) -x(1) -x(2); % 约束
```

```
g(2,1) = -(1)*x(2) -10;
```

第二步：在命令窗口调用有约束非线性优化函数 constr

```
x0 = [-1,1]; % 设置初始解向量
x = constr(  un? x0)
```

经过 29 次函数调用后，得到如下结果：

```
x =
9.5474 1.0474
```

极值点处的函数值和约束条件的值为：

```
[f,g] = fun(x)
f =
0.0236
g =
1.0e-15*
0.8882
0
```

可见，使用 Matlab 的优化工具箱解决优化问题简洁、明了，用户完全可将精力集中于需要解决的问题，而不需考虑各种优化算法的具体实现。

4.2.2 约束方程的规范化

由于 Matlab 的优化工具箱仅仅支持形如 $G(x)\leqslant 0$ 的约束方程形式以及变量的上下界约束。因此，对于非规范型的约束方程，必须进行变换。

对于形如 $g_i(x)\geqslant 0$ 的约束，可以将其等价为 $-g_i(x)\leqslant 0$。例如上例中的约束

$$x_1x_2 \ge -10$$

规范化为

$$-x_1x_2 - 10 \le 0$$

对于等式约束，Matlab 要求必须将等式约束方程置于约束变量 g 的前几个元素中，并在优化参数设置选项 options 向量中设置 options(13)为等式约束方程的个数。对于 options 向量的详细用法见下节。

例 2 仍以例 1 的优化问题为例，并加入以下等式约束

$$x_1 + x_2 = 1$$

将其改为规范等式约束

$$x_1 + x_2 - 1 = 0$$

求解过程

第一步：为目标函数即约束方程编写 M 文件——fun.m

```
function [f,g] = fun(x)
f = exp(x(1))*(4*x(1)^2+2*x(2)^2+4*x(1)*x(2)+2*x(2)+1);
g(1) = x(1)+x(2)-1; % 首先是等式约束
g(2) = 1.5+x(1)*x(2)-x(1)-x(2); % 不等式约束
g(3) = -x(1)*x(2)-10;
```

第二步：在命令窗口调用有约束非线性优化函数 constr

```
x0 = [-1,1]; % 设置初始解向量
options(13) = 1; % 有一个等式约束
x = constr(  un? x0,options)
```

经过 22 次函数调用后，得到如下结果：

```
x =
2.7016 3.7016
```

极值点处的函数值和约束条件的值为：

```
[f,g] = fun(x)
f =
1.6775
g =
0.0000 -9.5000 0.0000
```

对于变量的上下界约束，Matlab 通过优化函数的有界语法调用来实现。例如，对于 constr 函数，其有界语法调用格式为

```
x = constr(  un? x0,options,vlb,vub);
```

该调用将 x 限制在 vlb≤x≤vub。

例 3　仍以例 1 的优化问题为例，并将 x 限制为大于等于 0。

第一步：为目标函数即约束方程编写 M 文件——fun.m

与例 1 相同。

第二步：在命令窗口调用有约束非线性优化函数 constr

```
x0 = [-1,1];           % 设置初始解向量
options = [];          % 使用缺省设置
vlb = [0,0];           % 设置下界约束
vub = [];              % 无上界约束
x = constr(  un? x0,options,vlb,vub)
```

经过 10 次函数调用后，得到如下结果：

```
x =
0 1.5000
```

极值点处的函数值和约束条件的值为：

```
[f,g] = fun(x)
f =
8.5000
```

```
g =
0

                              10
```

在该例中，x 没有上界，因此 vub 被置为空矩阵。

当 vlb 或者 vub 的元素个数比向量 x 的元素数目少时，只有 x 的前几个元素被约束为有界。

上下界约束也可以用不等式约束来表示，具体表示方法为

上界：$x_i \leqslant U_B$　表示为 $x_i - U_b \leqslant 0$

下界：$x_i \geqslant U_B$　表示为 $-x_i + U_b \leqslant 0$

对于目标函数，Matlab 的优化函数用于处理求取最小值的情况，对于求取最大值的目标函数需要进行转换。

4.2.3　参数设置与附加参数传递

优化问题求解时常常需要对相对误差、使用算法等进行设置。考虑到对优化问题求解的灵活性，Matlab 提供了 options 向量来对优化函数进行参数设置。options 向量由 18 个元素，其中包括在实际调用中用户需要设置的所有优化参数。下表给出 options 向量中每个参数的意义。

表 4.2.1　控制参数列表

序 号	功　　能	缺 省 值	功 能 说 明
1	控制显示	0	在优化过程中控制输出信息，0 表示不显示中间结果；1 表示中间结果的列表显示；-1 表示隐藏警告信息
2	控制 x 精度	1e-4	自变量 x 的最低精度终止判据。当所有的终止判据都满足时，优化将终止
3	控制 f 精度	1e-4	目标函数 f 的最低精度终止判据。当所有的终止判据都满足时，优化将终止
4	控制 g 精度	1e-7	约束 g 的的最低精度终止判据。当所有的终止判据都满足时，优化将终止
5	算法控制	0	选择主要优化算法
6	SD 算法控制	0	选择搜索方向算法
7	搜索算法控制	0	选择线性搜索算法
8	函数值	N/A	算法结束时极值点的函数值，对 attgoal 和 minimax 而言，它包含一个到达因子。
9	梯度检查控制	0	当置为 1 时，在最初的几个迭代周期，梯度将与由有限插分计算的结果比较，此时梯度函数必须存在
10	函数计算计数	N/A	函数计算计数器
11	梯度计算计数	N/A	函数梯度计算或有限插分梯度计算的计数器
12	限定计数	N/A	限定函数梯度计算或有限插分梯度计算的次数
13	等式约束个数	0	等式约束条件的个数。等式约束必须放置在约束变量 g 的前几个元素中
14	最大迭代次数	0*n	最大迭代次数。该值缺省的被置为 n 的 100 倍，n 为自变量 x 的个数。在 fmins 中，缺省为 200n；在 fminu 中，缺省为 500n
15	目标数	0	尽可能接近 goals 的目标数，由函数 attgoal 使用
16	最小摄动控制	1e-8	有限插分梯度计算中变量的最小变化。对函数梯度计算而言，实际使用的摄动将自动调整以提高精度，它将在最小摄动和最大摄动之间变化
17	最大摄动控制	0.1	有限插分梯度计算中变量的最大变化
18	步长控制	N/A	步长参数。在第一次迭代中它常被赋值为 1 或更小

如果在对某优化函数进行调用时，没有使用 options 向量或者 options 向量为空向量，则会自动产生一个 options 向量并使用一组缺省值。如果需要对 options 向量中的某些元素重新赋值，则可首先通过 foptions 函数来产生一个 options 向量，然后对需要赋新值的元素进行赋值，其他元素仍然为缺省值。

1　foptions

功能：设置优化参数，显示参数值。

格式：help foptions

options = foptions

说明：在计算优化问题时，优化工具箱提供 options 向量，该向量用于设置优化算法参数和返回算法结束的状态。同时为保持一致性，options 向量对于大多数优化函数具有同样的意义。

help foptions 显示 option 向量中每个参数的意义和它的缺省值。

options = foptions 常用于优化算法结束后返回算法结束的状态。

options 中的一些参数可能根据问题的大小而定。而另外一些参数用来返回优化函数的返回信息，这些用于返回调用结果信息的参数没有缺省值。在表 4.2.1 中以 N/A 表示(N/A——Not Applicable)；还有一些参数具体地与使用的优化函数相关，这些参数将在后面章节的相关内容中介绍。

例 2　改变缺省设置的例子。

仍以例 1 的优化问题为例，现改变自变量及目标函数的终止条件。

第一步：为目标函数即约束方程编写 M 文件——fun.m

与例 1 相同。

第二步：在命令窗口调用有约束非线性优化函数 constr

```
x0 = [-1,1]; % 设置初始解向量
options = foptions
options(1) = 1; % 显示中间结果
options(2) = 1e-8; % 设置 x 终止条件
options(3) = 1e-8; % 设置 fun(x) 终止条件
x=constr(  un? x0,options)
f-COUNT   FUNCTION     MAX{g}      STEP    Procedures
    3      1.8394         0.5       1
    6      1.85127    -0.0919699    1   Hessian modified twice
    9     0.300167      9.32996     1
   12     0.529834     0.920855     1
   16     0.186965     -1.51704    0.5
   19    0.0729085     0.331311     1
   22    0.0353322   -0.0330268     1
   25    0.0235566   0.00318399     1
```

```
    28   0.0235504  9.03246e-008    1   Hessian modified
    29   0.0235504            0     1   Hessian modified
Optimization Converged Successfully
Active Constraints:
1
2
x =
9.5474   1.0474
```

同时在 Matlab 的优化工具箱中，为了用户程序的灵活性，避免太多的全局变量。每个优化函数都允许在函数调用的固定参数结尾,使用附加参数来直接将要使用的变量传递给 M 文件函数。需要注意的是，附加参数不能超过 10 个。如果目标函数 fun 和梯度函数 grad 定义为具有附加参数的格式

```
f = fun(x,p1,p2, ... )
df = grad(x,p1,p2 ... )
```

则在调用 fsolve 函数采用如下格式将传递这几个附加参数给目标函数 fun 和梯度函数 grad

```
fsolve(  un? x0,options,  rad? p1,p2, ... )
```

4.2.4 表达式优化

优化工具箱中的函数还能直接对由表达式描述的简单函数进行优化。这样，简单函数就可以不需要编写 M 文件而直接放在字符串中进行优化计算。如果被计算的函数变量是包含非希腊字母及数字的字符串(如+、-、* 等)，它将以一个表达式而不是以函数进行计算。

注意，当书写这种表达式时，自变量必须以小写字母 x 表示。

例 3 表达式优化的例子。

```
x = fminu(  in(x)? 1) % 求 sin(x)的最小值，初始值为 1
x = fsolve(  *x*x-[1,2;3,4]? ones(2,2)) %矩阵方程求解
x = leastsq(  *x-[3 5;9 10]? eye(2,2))%最小方差问题
```

附加参数(具有变量名 p1, p2, ？ 传递也可用于表达式优化中。

```
x = attgoal(  ort(eig(P1+P2*x*P3))? zeros(2,2),?
[-5,-3,-1],[5, 3, 1], [ ],-4*ones(2),4*ones(2),[ ],A,B,C);
```

这里，函数参数 p1, p2 和 p3 分别被置为等于变量 A, B, C。

4.3　线 性 规 划

4.3.1　线性规划概述

线性规划是优化理论发展最成熟，应用最广泛的一个分支。它讨论的是在一定的线性约束下对一个线性的目标函数求极值的问题。线性规划的标准形式可以写为：

目标函数

$$\min c_1x_1 + c_2x_2 + \Lambda + c_nx_n$$

约束

$$\begin{aligned} a_{11}x_1 + a_{12}x_2 + \Lambda + a_{1n}x_n &= b_1 \\ a_{21}x_1 + a_{22}x_2 + \Lambda + a_{2n}x_n &= b_2 \\ &\Lambda \\ a_{m1}x_1 + a_{m2}x_2 + \Lambda + a_{mn}x_n &= b_m \\ x_i \ge 0, \quad i = 1,2,\Lambda\ n \end{aligned} \tag{4-2-1}$$

也可以表示成矩阵形式

目标函数

$$\min \quad C^T X$$

约束

$$\begin{aligned} AX &= B \\ X &\ge 0 \end{aligned}$$

其中：$C=(c_1, c_2, \ldots, c_n)^T$, $A=(a_{ij})_{m*n}$ 和 $B=(b_1, b_2, ?\quad b_m)^T$ 为已知系数，$X=(x_1, x_2, \ldots,\ x_n)^T$ 为决策变量。

在线性规划的标准形式中，所有决策变量 x_i 均假定为非负的。在实际问题中这一假定通常是成立的。倘若不然，也可将其化为符合这一假定的等价的线性规划。例如，若变量 x_l 没有非负性假设，我们可用 $x'_i-x''_i$ 取代它，其中 x'_i 和 x''_i 是两个新的变量，且 $x'_i>0, x''_i>0$。

如果约束中存在不等式约束，则对于含≤的约束，可在约束左边加上一个非负变量使其称为等式约束；对于含≥的约束，可在约束左边减去一个非负变量使其称为等式约束。在约束中新增加的变量称为是松弛变量，原来的变量称为结构变量。

一个解 x 称为线性规划(4-2-1)的可行解，如果 x 满足 $Ax=B$，且 $x\geqslant 0$。由所有可行解构成的集合称为可行集。一个可行解 x^* 称为线性规划(4-3-1)的最优解，如果对所有的可行解 x，有 $C^Tx\leqslant C^Tx^*$ 成立。

欧氏空间 R^n 的子集 S 称为凸的，如果 S 中任意两点的连线也在 S 中，即 S 是凸的当且仅当对任意的 x_1, $x_2\in S$ 和 $\lambda\in[0, 1]$，有 $\lambda x_1+(1-\lambda)x_2\in S$。可以证明线性规划的可行集永远是

凸的。

一个点 x 成为凸集 S 的顶点，如果 $x\in S$，且 x 不能表示成其他任何两点的凸组合。已经证明，在可行集有界的情况下，线性规划的最优解一定是可行集中的一个顶点。这正是 Dantzig 给出的求解一般线性规划问题的算法——单纯形法的理论基础。简单地说，单纯形法仅仅检查可行集的顶点，而不是所有的可行解。首先，单纯形法选择一个顶点作为初始点，然后通过选择另一个顶点以改善目标函数值。重复以上过程直到目标函数值不能再改进，即可得到最优解。

对于解决大规模的或者特殊结构的线性规划问题，一些更先进的算法则更有效，如修正单纯形法、对偶单纯形法、原始对偶单纯形法、Wolfe-Dantzig 分解法以及 Karmarkar 内点法等。关于算法的详细说明请见相关书籍或者文献。

4.3.2 lp 函数

在优化工具箱中，lp 函数用于解线性规划问题。该函数使用单纯形法进行计算。通过解一个辅助的线性规划问题，来获取初始基本可行解。

- lp

功能：求解线性规划问题。

格式：
```
x = lp(c, A, b)
x = lp(c, A, b, vlb)
x = lp(c, A, b, vlb, vub)
x = lp(c, A, b, vlb, vub, x0)
x = lp(c, A, b, vlb, vub, x0, neqcstr)
x = lp(c, A, b, vlb, vub, x0, neqcstr, display)
[x,lambda,how] = lp(c,A,b, ... )
```

说明：lp 函数用于求解下述线性规划问题。

目标函数

$$\min_{x} c^T x$$

约束条件

$$Ax \le b$$

其中：A 为矩阵，b, c 为向量。

x = lp(c, A, b) 求解上述线性规划问题。返回线性规划的解向量 x。

x = lp(c, A, b, vlb, vub) 设置解向量的上下界，即解向量必须满足 vlb<=x<=vub。

x = lp(c, A, b, vlb, vub, x0)设置初始解向量 x0。

x = lp(c, A, b, vlb, vub, x0, neqcstr) 设置在约束中的等式约束的个数。等式约束必须位于约束方程的前面几个。

x = lp(c, A, b, vlb, vub, x0, neqcstr, display) 设置警告信息的显示。

[x, lambda] = lp(c, A, b, ...))同时返回拉格朗日(Lagrange)乘子 lambda。

[x, lambda, how] = lp(c, A, b, ...) 同时返回在最后一次调用过程中的错误状态。该错误状态字符串由变量 how 返回。

其中

c 为线性规划目标函数的系数向量。

A, b 为线性规划约束方程的系数。等式约束的系数必须位于矩阵 A 的前几行和向量 b 的前几个元素。

vlb 为下界约束，vub 为上界约束。一般地，vlb 和 vub 具有和 x 同样的大小。如果 vlb 由 n 个元素且比 x 的元素少，则只有 x 的前 n 个元素具有下界约束。vub 变量也遵守同样的原则。

x0 为初始解向量。缺省地，x0=zeros(size(x))。设置初始解向量可以使运算快速收敛。对于一个病态问题，好的初始解向量可以得到一个更好的解。

neqcstr 为等式约束的个数。

display 为控制警告信息显示的标志位。缺省的 display=0，即显示警告信息。如果 display=−1，则不显示警告信息。

lambda 为解的拉格朗日(Lagrange)乘子向量。拉格朗日乘子向量的长度等于 length(b)+length(vlb)+length(vub)，且元素顺序对应为 A,vlb,vub。

how 为求解过程中的错误状态指示字符串。如果 how='infeasible'，则该问题无可行解；如果 how='unbounded'，则该问题有无界解；如果 how='dependent'，则该问题的约束中有相关的等式约束，在求解过程中相关约束已经去除；如果 how='ok'，则该问题正常结束。

和其他优化工具箱中的函数一样，空的调用系数将导致调用过程中使用缺省值。例如：`lp(f, A, b, [ ], [ ], [ ], length(b))`表明该问题为一个等式约束问题，无上下界约束并使用缺省的初始解向量。

举例：求下述线性规划问题。

目标函数：$f(x)=-5x_1-4x_2-6_3$

约束方程：$x_1-x_2+x_3\leqslant 20$

$3x_1+2x_2+4x_3\leqslant 42$

$3x_1+2x_2\leqslant 30$

$0\leqslant x_1, 0\leqslant x_2, 0\leqslant x_3$

第一步：输入系数

```
c = [-5,-4,-6]
a = [1 -1 1
     3 2 4
     3 2 0];
b = [20; 42; 30];
```

第二步：求解

```
[x,lambda] = lp(c,a,b,zeros(3,1))
```

解为：

```
x =
0 15.0000 3.000
lambda =
0
1.5000
0.5000
1.0000 0 0
```

拉格朗日乘子(lambda)的前三个元素与非等式约束相关，lambda 的非零元素表示对应的约束为有效约束。在本问题中，第 2、3 个线性不等式约束为有效约束。lambda 的后面三个元素与下界约束相关，在本问题中，x_1 的下界约束为有效约束。

注意：1. 当问题无可行解时将给出以下信息

Warning: The constraints are overly stringent;

there is no feasible solution.

即约束条件过于严格。此时，lp 将给出一个对约束的破坏影响最小的解。

2. 当等式约束之间不一致时将给出以下信息

Warning: The equality constraints are overly stringent;

there is no feasible solution.

3. 当问题具有无界解时将给出以下信息

Warning: The solution is unbounded and at infinity;

the constraints are not restrictive enough.

4.4 非线性规划

4.4.1 无约束规划

1 无约束规划概述

无约束非线性问题的一般描述

目标函数

$$\min_{x} f(x)$$

其中：$x=[x_1,x_2,\ldots x_n]$为向量，$f(x)$为返回一标量值的目标函数。

无约束非线性规划有许多种算法。按照是否使用函数的导数信息来选择搜索方向，可将各种搜索算法分为 3 类：直接搜索类方法仅仅利用目标函数的函数值来确定搜索方向，这类方法对于目标函数不连续的情况非常适合；梯度类方法在目标函数具有一阶导数时非常有

效；而高阶类方法仅仅适用于目标函数的二阶导数可以计算的情况。

搜索算法的基本思想

1. 选定初始点 X^0，令 k=0。

2. 得到 x^k 后，选取一个搜索方向 P^k，使得沿这个方向目标函数 $f(x)$的值可以下降。

3. 由 x^k 出发，沿 P^k 方向选取步长 λ_k，使得 $f(x^k+\lambda_k P^k)<f(x^k)$，由此得到下一个点 $x^{k+1}=x^k+\lambda_k P^k$。

4. 检验 x^{k+1} 是否是满足精度要求的最优解。如果是，则结束计算；如果不是，则返回 2，并令 $k=k+1$，继续迭代。

可见，搜索算法的关键在于搜索方向及搜索步长的选取。

一维搜索算法(步长选取)

步长选取实际上是对单变量函数 $\Phi(\lambda)=f(x^k+\lambda_k*P^k)$求极小值的问题。常用的一维搜索算法包括黄金分割法、Fibonacci 法、抛物线插值法和三次插值法。

黄金分割法是一种等速对称的一维搜索算法，它采用将试点取在区间的黄金分割点上来达到缩短搜索区间的目的。从而寻求函数 $\Phi(\lambda)$的极小点。

Fibonacci 法通过构造 Fibonacci 数来缩短搜索区间，从而寻求函数 $\Phi(\lambda)$的极小点。

抛物线插值法的基本思想是利用二次函数：$P(x)=a_0+a_1x+a_2x^2$ 的极小点去逼近 $f(x)$的极小点。

三次插值法则是通过 a、b 两点处的函数值 $f(a)$、$f(b)$和导数值 $f'(a)$、$f'(b)$来构造三次插值多项式 $P(x)=a_0+a_1x+a_2x^2+a_3x^3$，以 $P(x)$的极小点作为 $f(x)$的极小点。在容易求得 $f(x)$的导数值时，三次插值法十分有效。

搜索方向的确定

一般地，对无约束的非线性规划算法的划分就是根据不同的搜索方向选取方法来确定的。下面给出几种不同的算法简介。

最速下降法的基本思想是：当给定某点 X^k 后，如果它不是极小点，则寻找 X^k 处函数 $f(X)$下降最快的方向作为搜索方向。由于函数 $f(x)$沿负梯度方向下降最快，故选 $P^k=-\nabla f(x^k)$，$\nabla f(x^k)$为 $f(x)$在 X^k 处的梯度。

牛顿法的搜索方向选取为：$P^k=-H(x^k)^{-1}*\nabla f(x^k)$，即在函数的负梯度方向上增加一个修正矩阵，该修正矩阵即为目标函数的二阶导数矩阵(Hessian 矩阵)的逆矩阵。

由于牛顿法中修正矩阵直接计算较难，从而又发展了各种拟牛顿(Quasi-Newton)方法，如常用的 BFGS(Broyden-Flether-Goldfarb-Shanno)方法、DFP(Davidon-Flether-Powell)方法等。其中 BFGS 方法采用下述公式来计算目标函数的 Hessian 矩阵的逆矩阵$-H(x^k)^{-1}$，在公式中

$$\overline{H}_k=-H\left(x^k\right)^{-1}$$

$$\overline{H}_{k+1}=\overline{H}_k+{q_k q_k^T}\Big/{q_k^T s_k}-{\overline{H}_k^T s_k^T s_k \overline{H}_k}\Big/{s_k^T \overline{H}_k s_k}$$

其中

$$s_k=x_{k+1}-x_k$$

$$q_k=\nabla f\left(x_{k+1}\right)-\nabla f\left(x_k\right)$$

将上式右边第三项中的 S_k 以 q_k 代替，即可得到 DFP 方法的计算公式。

2 fminu、fmins 函数

在优化工具箱中，fminu、fmins 函数用于解线性规划问题。可以选择使用 BFGS 公式来逼近 Hessian 矩阵的拟牛顿(quasi-Newton)方法；或者采用 DFP 公式来逼近 Hessian 矩阵的拟牛顿(quasi-Newton)方法；或者采用最速下降法。线性搜索算法可以选择使用一种二次和三次多项式插值的混合算法；或者使用三次多项式插值算法。

- fminu, fmins

功能：求解无约束非线性最优化问题。

格式：

```
x = fminu('fun', x0)
x = fminu('fun', x0, options)
x = fminu('fun', x0, options, 'grad')
x = fminu('fun', x0, options, 'grad', p1, p2,   ... )
[x, options] = fminu('fun', x0, ... )
[ ... ] = fmins('fun', x0, ... )
```

说明：x = fminu('fun', x0)求函数 fun 的最小值。fun 函数定义在 M 文件 fun.m 中，并置初始解向量为 x0。

x = fminu('fun', x0, options) 设置可选参数的值而不是采用缺省值。可选参数在 options 向量中设置。options 的详细用法见 4.2.3。

x = fminu('fun', x0, options, 'grad') 使用由函数 grad 计算的梯度信息。函数 grad 在 M 文件 grad.m 中定义。缺省地，梯度将通过有限差分方法计算目标函数的导数来获得。

x = fminu('fun', x0, options, 'grad', p1, p2, ...) 传递附加的参数 p1，p2 等。附加参数的详细用法见 4.2.3。

[x, options] = fminu('fun', x0, ...) 同时返回在计算中使用的一些参数，例如 options[10]包含了计算过程中目标函数的计算次数。其中

x_0 为初始解向量。

fun 为包含目标函数的函数名字符串。fun 函数返回 1 个参数：目标函数的函数值 f。f 为标量，并具有 [f] = fun(x) 格式。也可以用字符串表达式达到同样的目的，例如 `x = fminu(   in(x.*x)? x0)`。

options 为参数控制向量。在 options 的 18 个元素中，fminu 函数使用的输入参数包括：1, 2, 3, 6, 7, 9, 14, 16, 17；fminu 函数使用的输出参数包括：8, 10, 11, 18；fmins 函数使用的输入参数包括：1, 2, 3, 14；fmins 函数使用的输出参数包括：8, 10。其中

options(1)控制显示。设置为 1 将给出中间结果的列表显示。

options(2)控制 x 的精度。

options(3) 控制目标函数 f 的精度。

当 options(2)、options(3)参数全部满足后，算法将结束。

对 options(6),、options(7)的讨论见后面算法部分。

对 options 的详细讨论请见 4.2.3。

grid 为包含梯度函数的函数名字符串，该函数具有 df = grad(x)格式。其中 df 为向量，它是目标函数 f 对于向量 x 的偏微分。要注意的是由于 fmins 函数不使用梯度信息，该参数被忽略。

p1,p2,…为传递给 fun 和 grid 的附加参数。附加参数的详细用法见 4.2.3。

举例：求无约束非线性问题

$$f(x) = 100\left(x_2 - x_1^2\right)^2 + \left(1 - x_1\right)^2$$

初始解向量：x=[−1.2　1]

第一步：编写 M 文件

```
function f = fun(x)
f = 100*(x(2)-x(1)^2)^2+(1-x(1))^2;
```

第二步：求解

```
x = [-1.2,1]
x = fminu(  un? x)
x =
1.0000 1.0000 fun(x) = 8.8348e-11
```

算法：对 fminu 函数优化算法的控制通过 options(6) 参数实现，缺省地 options(6)=0。此时使用拟牛顿(quasi-Newton)方法，这种方法采用 BFGS 公式来逼近 Hessian 矩阵；如果设置 options(6)=1，则采用 DFP 公式来逼近 Hessian 矩阵；如果设置 options(6)=2，则采用最速下降法。

对 fminu 函数的线性搜索算法的控制通过 options(7)参数实现。缺省地 options(7)=0，使用一种二次和三次多项式插值的混合算法；如果 options(7)=1，使用三次多项式插值算法。该算法需要较少的函数计算和较多的梯度计算。

fmins 函数使用一种单纯型搜索算法，这是一种直接法而不是像 fminu 函数那样使用解析法。

如果目标函数大于 2 阶，则一般地 fmins 函数不如 fminu 函数；但对于非常不连续的函数，则 fminu 函数不如 fmins 函数更具鲁棒性。

注意：1. 对于 fminu 函数，目标函数必须连续。fmins 函数常用来处理不连续的情况。这两个函数有可能给出局部最优解。

2. 对于目标函数中存在平方和的情况，不应该使用 fminu 和 fmins 函数，而应该使用 leastsq 函数。

4.4.2 二 次 规 划

如果非线性规划的目标函数为自变量 x 的二次函数，约束条件又全是线性的，这种规划就是二次规划。

二次规划的一般描述

目标函数

$$\min_{x} \frac{1}{2} x^T H x + c^T x$$

约束条件

$$Ax \le b$$

其中：H, A 为矩阵，且 H 为对称矩阵。c, b 为向量。

对于二次规划的求解具有很多方法，如可行方向法、制约函数法、又称序列无约束优化方法(SUMT——Sequential Unconstrained Minimization Technique)以及有效集法(Active Set Method)等。其中有效集法是近年来发胀的一种通用的二次规划算法。关于该算法详见参考文献[3]。

4.4.3 有约束规划

1 约束规划概述

约束非线性规划的一般描述

目标函数

$$\min_{x} f(x)$$

约束条件

$$G(x) \le 0$$

其中：$x=[x_1\ x_2\ ?\ \ x_n]^T$ 为向量，$G(x)=[g_1(x)\ g_2(x)\ ?\ \ g_m(x)]^T$ 为函数向量，$f(x)$为标量函数，$f(x)$和 $G(x)$均可为非线性函数，$G(x)$既可以为等式约束也可以为不等式约束。

对于约束非线性规划问题，已经建立了大量的计算方法，如可行方向法、梯度投影法、罚函数法、线性近似法等，详见参考文献[2]。但这些算法均仅仅能解决一类特殊的非线性规划问题。

在理论上，Kuhn-Tucher 条件则具有很重要的意义。

对于一个不等式约束 $g_i(x)\leqslant 0$，如果 $g_i(x^*)=0$，则称该约束在点 x^*处为起作用约束；设 x^*为问题的一个可行解，如果约束的梯度向量$\nabla g_i(x^*)$线性无关，x^*为局部最优解，则必然存在不全为 0 的数 $u_i(i=1,\ldots,m)$，使得以下 Kuhn-Tucker 条件成立：

$$\begin{cases}\nabla f(x^*)+\sum_{i=1}^{m}\lambda_i\nabla g_i(x^*)=0\\ \lambda_i g_i(x^*)=0 \quad i=1,2,\Lambda\ ,m\\ \lambda_i\ge 0 \qquad i=1,2,\Lambda\ ,m\end{cases}$$

如果 $f(x)$和 $g_i(x)$ (i=1,?　m)均为凸的，并且可导，x^*满足 Kuhn-Tucker 条件，则 x^*为全局最优解。

目前很多算法都集中在对 Kuhn-Tucker 方程的求解上,这些方法都是直接计算 Lagrange 乘子。

SQP 算法(序列二次规划 Sequential Quadratic Programming)即将原问题化为一系列的二次规划问题进行求解。它通过使用拟牛顿更新程序来对 Kuhn-Tucker 方程累积二阶信息，以保证超线性的收敛。

对上述非线性约束优化问题，SQP 算法的基本思想是对下述 Lagrange 函数的二次近似求解 QP 子问题：

$$L(x,\lambda)=f(x)+\sum_{i=1}^{m}\lambda_i g_i(x)$$

线性化非线性约束条件后可得到 QP 子问题：

目标函数

$$\min \tfrac{1}{2}d^T H_k d+\nabla f(x_k)^T d$$

约束

$$\nabla g_i(x)^T d+g_i(x)=0 \qquad i=1,\Lambda\ ,m_e$$
$$\nabla g_i(x)^T d+g_i(x)\le 0 \qquad i=m_e+1,\Lambda\ ,m$$

该问题可以通过任何 QP 算法进行求解。

SQP 实现

Matlab 优化工具箱的 SQP 算法实现由如下 3 个部分组成。

(1)　更新 Lagrange 函数的 Hessian 矩阵

在每一次迭代中，H 均作为 Lagrange 函数的 Hessian 矩阵的正定拟牛顿近似。并通过 BFGS 方法进行计算。

Hessian 矩阵的 BFGS 更新

$$H_{k+1}=H_k+{q_k q_k^T}\big/{q_k^T s_k}-{H_k^T H_k}\big/{s_k^T H_k s_k}$$

其中

$$s_k = x_{k+1} - x_k$$
$$q_k = \nabla f(x_{k+1}) + \sum_{i=1}^{m} \lambda_i \nabla g_i(x_{k+1}) - \left(\nabla f(x_k) + \sum_{i=1}^{m} \lambda_i \nabla g_i(x_k) \right)$$

(2) 二次规划问题求解

SQP 方法的每一次主迭代都要求解一个 QP 问题。

目标函数

$$\min \tfrac{1}{2} x^T H x + c^T x$$

约束方程

$$A_i x = b \qquad i = 1, \Lambda\ , m_e$$
$$A_i x \le b \qquad i = m_e + 1, \Lambda\ , m$$

(3) 一维搜索和目标函数计算

一维搜索可采用合适的搜索算法进行计算。要注意的是 SQP 算法的初始化，如果从 SQP 方法中得到的当前计算点不合适，则可通过求解线性规划问题得到初始值。

目标函数

$$\min_{\gamma \in R, x \in R^n} \gamma$$

约束方程

$$A_i x = b \quad i = 1, \Lambda\ , m_e$$
$$A_i x - \gamma \le b \quad i = m_e + 1, \Lambda\ , m$$

其中 A_i 表示矩阵 A 的第 i 行。

2 fmin 函数——标量最优求解

标量最优问题的一般描述

目标函数

$$\min_{a} f(a)$$

其中：a 为标量，$f(a)$为一个返回标量值的函数。

在优化工具箱中，fmin 函数用于解标量最优问题。而且该函数允许设置变量的上下解约束。

- fmin

  功能：求解区域约束单变量问题。

  格式：a = fmin('fun', a1, a2)

  a = fmin('fun', a1, a2, options)

a = fmin('fun', a1, a2, options, p1, p2, ...)

[a, options] = fmin('function', a1, a2, ...)

说明：区域约束单变量问题的一般描述

目标函数

$$\min_{a} f(a)$$

约束条件

$$a_1 < a < a_2$$

其中：a, a_1, a_2 为标量，$f(a)$为一个返回标量值的函数。

a = fmin('fun', a1, a2)求函数的局部最优解。目标函数定义在名为 fun.m 的 M 文件中，且其解必须满足 a1≤a≤a2。

a = fmin('fun', a1, a2, options) 设置可选参数的值而不是采用缺省值。可选参数在 options 向量中设置。options 的详细用法见 4.2.3。

a = fmin('function', a1, a2, options, p1, p2, ...) 传递附加的参数 p1，p2 等。附加参数的详细用法见 4.2.3。

[a,options] = fmin('fun', a1, a2, ...) 同时返回在计算中使用的一些参数。例如 options[10]包含了计算过程中目标函数的计算次数。其中

fun 为包含目标函数的函数名字符串。fun 函数返回 1 个参数：目标函数的函数值 f，f 为标量，例如 `f = fun(x, xdata)`。也可以用字符串表达式达到同样的目的，例如 `a = fmin(  in(x*x)? a1,a2)`，a1、a2 为函数的区域约束。

options 为参数控制向量。在 options 的 18 个元素中，fmin 函数使用的输入参数包括 1、2、14，constr 函数使用的输出参数包括 8、10。其中

options(1)控制显示。设置为 1 将给出中间结果的列表显示。

options(2)控制 x 的精度。

options(14)控制函数的计算次数。

p1, p2, …为传递给 fmin 的附加参数。

举例：求下述标量函数在(0,5)区域内的最小值。

目标函数：$f=(a-3)^2-1$

%第一步：编写 M 函数

```
function f = fun(a)
f = (a-3).^2-1;
```

%第二步：求解

```
a = fmin(  un? 0,5)
a =
3
The value at the minimum is
y = f(a)
```

```
y =
     1
```

算法：该函数采用黄金分割和抛物线搜索算法。

注意：1. 该函数可能给出局部最优解。

2. 目标函数和约束条件必须为实型的。

3. 当解位于区域边界上时，该函数将具有较慢的收敛速度，此时采用 constr 函数将给出更快和更精确的解。

3 constr 函数

多变量非线性约束最优问题的一般描述

目标函数

$$\min_{x} f(x)$$

约束条件

$$G(x) \le 0$$

其中：x 为向量，$G(x)$为函数向量，$f(x)$为标量函数，$f(x)$和 $G(x)$均可为非线性函数，$G(x)$既可以为等式约束也可以为不等式约束。

在优化工具箱中，constr 函数用于求解一般约束最优问题。该函数使用 SQP 算法。

- constr

功能：多变量非线性约束最优问题求解。

格式：x = constr('fun', x0)

x = constr('fun', x0, options)

x = constr('fun', x0, options, vlb, vub, 'grad')

x = constr('fun', x0, options, vlb, vub, 'grad', p1, p2, ...)

[x, options] = constr('fun', x0, ...)

[x, options,lambda] = constr('fun', x0, ...)

[x, options, lambda, hess] = constr('fun', x0, ...)

说明：x = constr('fun', x0)求解非线性约束最优化问题。目标函数和约束条件定义在 M 文件中，文件名为 fun.m。初始解向量为 x0。

x = constr('fun', x0, options)设置可选参数的值而不是采用缺省值。可选参数在 options 向量中设置。options 的详细用法见 4.2.3。

x = constr('fun', x0, options, vlb, vub) 设置解向量的上下界，即解向量必须满足 vlb<=x<=vub。

x = constr('fun', x0, options, vlb, vub, 'grad')使用由函数 grad 计算的梯度信息。函数 grad 在 M 文件 grad.m 中定义。缺省地，梯度将通过有限差分方法计算目标函数的导数来获得。

x = constr('fun', x0, options, vlb, vub, 'grad', p1, p2, ...)传递附加的参数 p1, p2 等。附加参数的详细用法见 4.2.3 节。

[x, options] = constr('fun', x0, ...)同时返回在计算中使用的一些参数。例如 options[10]包含计算过程中目标函数的计算次数。

[x,options,lambda] = constr('fun', x0, ...) 同时返回拉格朗日(Lagrange)乘子 lambda。

[x, options, lambda, hess] = constr('fun', x0, ...)同时返回最后一次迭代时的 Hessian 矩阵值 hess。

其中

x_0 为初始解向量。

fun 为包含目标函数和约束条件的函数名字符串。fun 函数返回 2 个参数：标量目标函数 f 和向量约束条件，并具有 [f,g] = fun(x) 格式。也可以用字符串表达式达到同样的目的。例如：`x=constr('f = fun(x); g = cstr(x); ', x0)`，

vlb, vub 为上下界约束。vlb 为下界约束，vub 为上界约束。一般地，vlb 和 vub 具有和 x 同样的大小。如果 vlb 由 n 个元素且比 x 的元素少，则只有 x 的前 n 个元素具有下界约束。vub 变量也遵守同样的原则。

options 为参数控制向量。在 options 的 18 个元素中，constr 函数使用的输入参数包括 1、2、3、4、9、13、14、16、17，constr 函数使用的输出参数包括 8、10、11、18。其中

options(1)控制显示。设置为 1 将给出中间结果的列表显示。

options(2)控制 x 的精度。

options(3) 控制目标函数 f 的精度。

options(4)控制对约束的越限程度。

当 options(2)、options(3)、options(4)参数全部满足后，算法将结束。

grid 为包含梯度函数的函数名字符串，该函数具有[df, dg] = grad(x)格式。其中 df 为向量，它是目标函数 f 对于向量 x 的偏微分；dg 为矩阵，它是约束方程向量对于向量 x 的偏微分。

p1,p2,…为传递给 fun 和 grid 的附加参数。附加参数的详细用法见 4.2.3。

lambda 为解的拉格朗日(Lagrange)乘子向量。拉格朗日乘子向量的长度等于 length(g)+length(vlb)+length(vub)，且元素顺序对应为 A, vlb, vub。

hess 为采用 Quasi-Newton 逼近法时最后一次迭代的 Hessian 矩阵值。

举例：求解如下非线性约束最优问题。

目标函数：$f(x)=-x_1*x_2*x_3$

约束条件：$-x_1-2x_2-2x_3\leqslant 0$；$x_1+2x_2+2x_3\leqslant 72$

初始解向量：x=[10 10 10]

第一步：编写 M 文件

```
function [f,g] = fun(x)
f = -x(1) * x(2) * x(3);
g(1) = -x(1)-2 * x(2) -2 * x(3);
```

```
g(2) = x(1) + 2*x(2) + 2*x(3) -72;
```

第二步：求解

```
x0 = [10,10,10];
x = constr(  un? x0)
```

经过 49 次运算后，结果为

```
x =
24.0000 12.0000 12.0000
[f,g] = fun(x)
f =
                    3.4560e+03
g =
72  0
```

注意：1. 该函数可能给出局部最优解。

2. 如果问题无可行解，将给出一个对约束的破坏影响最小的解。
3. 目标函数和约束条件必须为实型的。
4. 当存在一致的相关等式约束时，算法将对等式约束去偶,并显示‘dependent’；如果存在不一致的等式约束，则算法无可行解，并显示‘infeasible’。

4.5 最小最大(minmax)问题

最小最大问题的一般描述

目标函数

$$\min_{x} \max_{\{F_i\}} \{F_i(x)\}$$

约束条件

$$G(x) \le 0$$

其中：x为向量，$F(x)$, $G(x)$为函数向量，$F_i(x)$为$F(x)$的第i个分量。

在优化工具箱中，minmax 函数用于求解一般约束最优问题。minmax 函数使用 SQP 方法，但对 SQP 的一维搜索方法和 Hessian 矩阵的更新进行了改进，以更适应这类问题。SQP 方法见 4.4.3 节关于该方法的详细说明。

- minimax

功能：求解最小最大问题。

格式：x = minimax('fun', x0)

x = minimax('fun', x0, options)

x = minimax('fun', x0, options, vlb, vub, 'grad')

x = minimax('fun', x0, options, vlb, vub, 'grad', p1, p2, ...)

[x,options] = minimax('fun', x0, ...)

说明：x = minimax('fun', x0)求解最小最大问题。目标函数和约束条件定义在 M 文件中，文件名为 fun.m。初始解向量为 x0。

x = minimax('fun', x0, options) 设置可选参数的值而不是采用缺省值。可选参数在 options 向量中设置。options 的详细用法见 4.2.3 节。

x = minimax ('fun', x0, options, vlb, vub) 设置解向量的上下界，即解向量必须满足 vlb<=x<=vub。

x = minimax('fun', x0, options, vlb, vub, 'grad') 使用由函数 grad 计算的梯度信息。函数 grad 在 M 文件 grad.m 中定义。缺省时，梯度将通过有限差分方法计算目标函数的导数来获得。

x = minimax('fun', x0, options, vlb, vub, 'grad', p1, p2, ...) 传递附加的参数 p1，p2 等。附加参数的详细用法见 4.2.3 节。

[x, options] = minimax('fun',x0, ...) 同时返回在计算中使用的一些参数。例如 options[10]包含了计算过程中目标函数的计算次数。其中

x_0为初始解向量。

fun 为包含目标函数和约束条件的函数名字符串。fun 函数返回 2 个参数：标量目标函数 f 和向量约束条件 g，并具有 [f, g] = fun(x) 格式。如果约束中存在等式约束，等式约束必须位于约束方程的前面几个，且必须将等式约束的个数传递给 options(13)。如果目标函数中要求某个元素的绝对值最小最大问题(例如：minmax abs{F(x)})，则应将这几个元素置于目标函数的前几个元素，并用 options(15)设置绝对值最小最大分量的个数。对 fun 函数也可以用字符串表达式来描述，例如 `x = minmax('f = fun(x); g = cstr(x); ', x0)`。

vlb 为下界约束，vub 为上界约束。一般地，vlb 和 vub 具有和 x 同样的大小。如果 vlb 由 n 个元素且比 x 的元素少，则只有 x 的前 n 个元素具有下界约束。vub 变量也遵守同样的原则。

options 为参数控制向量。在 options 的 18 个元素中，constr 函数使用的输入参数包括 1、2、3、4、7、9、13、14、15、16、17，constr 函数使用的输出参数包括 8、10、11、18。其中

options(1)控制显示。设置为 1 将给出中间结果的列表显示。

options(2)控制 x 的精度。

options(3) 控制目标函数 f 的精度。

options(4)控制对约束的越限程度。

当 options(2)、options(3)、options(4)参数全部满足后，算法将结束。

options(13)设置等式约束的个数。

options(15)设置绝对值最小最大分量的个数。

对 options(7)、options(8)的讨论见算法部分。

对 options 详细讨论请见 4.2.3 节。

grad 为包含梯度函数的函数名字符串，该函数具有[df,dg] = grad(x)格式。其中 df 为向量，它是目标函数 f 对于向量 x 的偏微分；dg 为矩阵，它是约束方程向量对于向量 x 的偏微分。

p1,p2,…为传递给 fun 和 grid 的附加参数。附加参数的详细用法见 4.2.3 节。

举例：(1) 求下述最小最大问题。

$$[f_1(x), f_2(x), f_3(x), f_4(x), f_5(x)]$$

其中

$$f_1(x) = 2x_1^2 + x_2^2 - 48x_1 - 40x_2 + 304$$
$$f_2(x) = -x_1^2 - 3x_2^2$$
$$f_3(x) = x_1 + 3x_2 - 18$$
$$f_4(x) = -x_1 - x_2$$
$$f_5(x) = x_1 + x_2 - 8$$

第一步：编写 M 文件

```
function [f,g] = fun(x)
f(1)=2*x(1)^2+x(2)^2-48*x(1)-40*x(2)+304;
f(2)= x(1)^2 -3*x(2);
f(3)= x(1) + 3*x(2) -18;
f(4)= -x(1)-x(2);
f(5) = x(1) + x(2) -8;
g = [ ]; % 无约束
```

第二步：求解

```
x0 = [0.1,0.1];
x = minimax(  un? x0)
```

经过 29 次运算后，结果为

```
x =
4.0000 4.0000
fun(x)
ans =
0.0000 -16.0000 -2.0000 -8.0000 0.000
```

(2) 求上述问题的绝对值最小最大问题。

即目标函数为

$$[abs(f_1(x)), abs(f_2(x)), abs(f_3(x)), abs(f_4(x)), abs(f_5(x))]$$

第一步：编写 M 文件(与例 1 相同)

第二步：求解

```
x0 = [0.1,0.1];
options(15) = 5; % 全部为绝对值最小最大分量
```

```
x = minimax(  un? x0,options)
```

经过 39 运算，解为

```
x =
8.7769 0.6613
fun(x)
ans =
10.7609 -7.2391 -9.4382 1.4382
```

4.6　半无限(Semi-infinite)问题

半无限约束非线性问题的一般描述

目标函数

$$\min_{x} f(x)$$

约束条件

$$G(x) \le 0$$
$$K(x,w) \le 0$$

其中：*x, w* 为向量，*G(x)*为函数向量，为问题的等式或者不等式约束。*K(x,w)*为半无限约束，通常 *w* 的长度为 2。

在优化工具箱中，seminf 函数用于求解半无限约束非线性问题。seminf 函数使用二次和三次混合插值法来估计半无限约束的峰值，从而形成 constr 函数使用的约束集，当约束改变时，重新分配 langrange 乘子向量。

- seminf

功能：求解半无限约束非线性问题。

格式：x = seminf('fun', n, x0)

x = seminf('fun', n, x0, options)

x = seminf('fun', n, x0, options, vlb, vub)

x = seminf('fun', n, x0, options, vlb, vub, p1, p2, ...)

[x, options] = seminf('fun', n, x0, ...)

说明：x = seminf('fun', n, x0) 求解半无限问题。目标函数和约束条件定义在 M 文件中，文件名为 fun.m，包括 n 个半无限约束。初始解向量为 x0。

x = seminf('fun', n, x0, options) 设置可选参数的值而不是采用缺省值。可选参数在 options 向量中设置。options 的详细用法见 4.2.3 节。

x = seminf('fun' ,n, x0, options, vlb, vub) 设置解向量的上下界，即解向量必须满足 vlb<=x<=vub。

x = seminf('fun', n, x0, options, vlb, vub, p1, p2, ...) 传递附加的参数 p1，p2 等。

附加参数的详细用法见 4.2.3 节。

[x, options] = seminf('fun', n, x0, ...) 同时返回在计算中使用的一些参数。例如 options[10]包含了计算过程中目标函数的计算次数。其中

x_0 为初始解向量。

fun 为包含目标函数和约束条件的函数名字符串。fun 函数具有[f,g,K1,K2, ...,Kn,s] = fun(x,s) 格式。f 为返回的目标函数值，g 为约束函数返回值，K1,K2, ...,Kn 为由向量 w 的采样值计算得到的半无限约束函数的返回值。矩阵 s 为向量 w 的采样值(详见举例)。如果约束中存在等式约束，等式约束必须位于约束方程的前面几个，且必须将等式约束的个数传递给 options(13)。

vlb 为下界约束，vub 为上界约束。一般地，vlb 和 vub 具有和 x 同样的大小。如果 vlb 由 n 个元素且比 x 的元素少，则只有 x 的前 n 个元素具有下界约束。vub 变量也遵守同样的原则。

options 为参数控制向量。在 options 的 18 个元素中，constr 函数使用的输入参数包括 1、2、3、4、9、13、14、16、17，constr 函数使用的输出参数包括 8、10、11、18。其中

options(1)控制显示。设置为 1 将给出中间结果的列表显示。

options(2)控制 x 的精度。

options(3) 控制目标函数 f 的精度。

options(4)控制对约束的越限程度。

当 options(2)、options(3)、options(4)参数全部满足后，算法将结束。

p1,p2,…为传递给 fun 和 grid 的附加参数。附加参数的详细用法见 4.2.3 节。

举例：(1) 一维的例子

目标函数

$$f(x)=(x_1-0.5)^2+(x_2-0.5)^2+(x_3-0.5)^2$$

约束方程

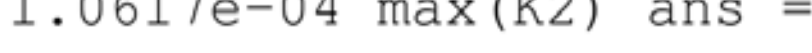
$$K_1(x_1,w_1)=\sin(w_1x_1)\cos(w_1x_2)-\frac{1}{1000}(w_1-50)^2-\sin(w_1x_3)-x_3\le 1$$
$$K_2(x_1,w_2)=\sin(w_2x_2)\cos(w_2x_1)-\frac{1}{1000}(w_2-50)^2-\sin(w_2x_3)-x_3\le 1$$
$$1\le w_1\le 100$$
$$1\le w_2\le 100$$

第一步：编写 M 文件

```
function [f,G,K1,K2,s] = fun(x,s)
% 初始化采样间距
if isnan(s(1,1)),s = [0.2 0; 0.2 0]; end
% 设置样本集
```

```
w1 = 1:s(1,1):100;
w2 = 1:s(2,1):100;
% 半无限约束
K1 = sin(w1*x(1)).*cos(w1*x(2))-1/1000*(w1-50).^2 ...
       sin(w1*x(3))-x(3)-1;
K2 = sin(w2*x(2)).*cos(w2*x(1))-1/1000*(w2-50).^2...

                            sin(w2*x(3))-x(3)-1;
% 无其他约束
G = [ ];
% 目标函数
f = sum((X-0.5).^2);
% 绘制，见图 4.6.1
plot(w1,K1,w2,K2),title(  emi-infinite constraints?
第二步：求解
x0 = [0.5,0.2,0.3]; % 设置初始值
x = seminf(  un? 2,x0)
x =
0.6956 0.3052 0.4261
[f,G,K1,K2] = fun(x,NaN);
  f =
  0.0817
  max(K1)
  ans =
```



```
              1.0617e-04 max(K2) ans =
  0.0023
```

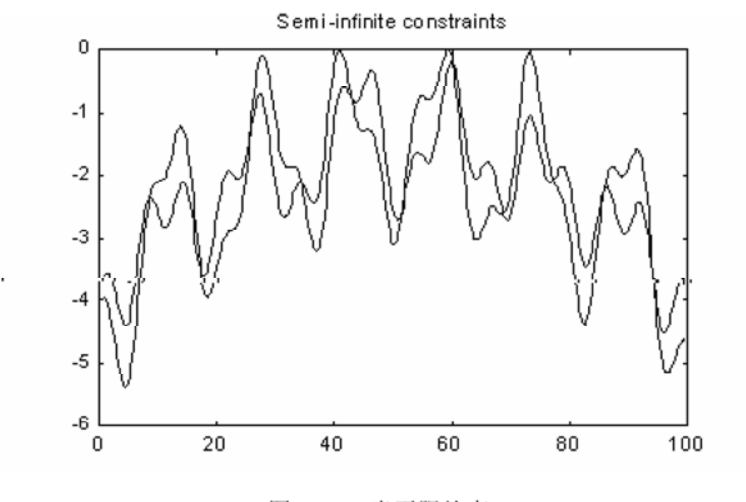


图 4.6.1　半无限约束

(2)　二维的例子

目标函数

$$f(x)=(x_1-0.5)^2+(x_2-0.5)^2+(x_3-0.5)^2$$

约束方程

$$K_1(x_1,w_1)=\sin(w_1x_1)\cos(w_2x_2)-\frac{1}{1000}(w_1-50)^2-\sin(w_1x_3)-x_3+$$
$$\sin(w_2x_2)\cos(w_1x_1)-\frac{1}{1000}(w_2-50)^2-\sin(w_2x_3)-x_3\le 1.5$$
$$1\le w_1\le 100$$
$$1\le w_2\le 100$$

初始解 x=[0.2, 0.2, 0.2]。

第一步：编写 M 文件

```
function [f,G,K1,s] = fun(x,s)
% 初始化采样间距
if isnan(s(1,1)),s = [2 2];
% 设置样本集
w1 = 1:s(1,1):100;
w2 = 1:s(2,1):100;
[wx,wy]=meshdom(w1,w2);
% 半无限约束
K1 = sin(wx*x(1)).*cos(wy*x(2)) -1/1000*(wx-50).^2 -...
sin(wx*x(3))-x(3)+sin(wy*x(2)).*cos(wx*x(1)) -...
1/1000*(wy-50).^2-sin(wy*x(3))-x(3)-1.5;
% 无其他约束
G = [ ];
% 目标函数
f = sum((x-0.5).^2);
% 绘制，见图 4.6.2
mesh(K1),title(  emi-infinite constraints?
```

第二步：求解

```
x0 = [0.2,0.2,0.2]; % 设置初始值
x=seminf(  un? 1,x0);
x=
0.2033 0.2034 0.1930
[F,G,K1]=fun(x,NaN);
max(max(K1))
ans=
0.0273
```

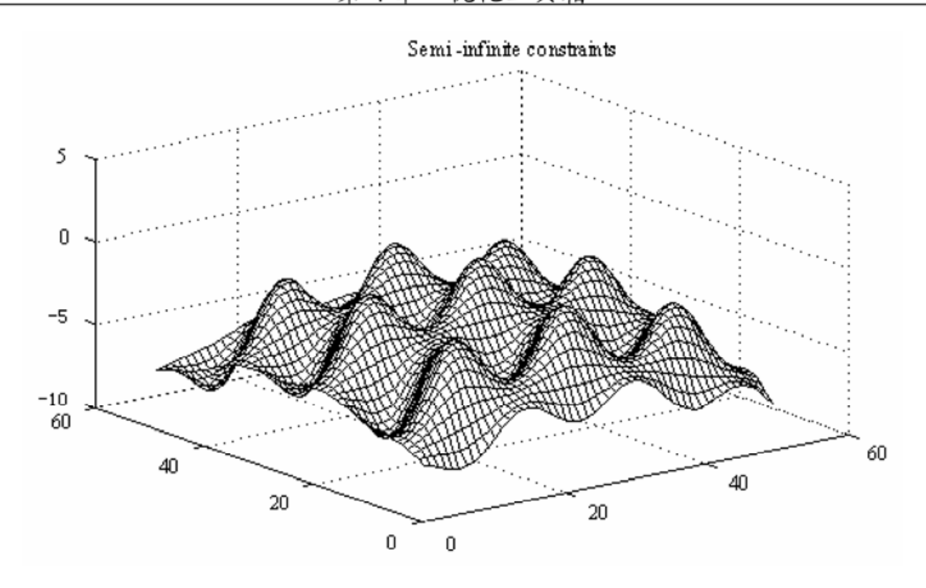


图 4.6.2　二维半无限约束

4.7　多目标(Goal Attainment)规划

多目标规划定义为在一组约束下，对多个不同的目标函数进行优化。

一般形式

目标函数

$$\min\left[f_1(x), f_2(x), \Lambda\ , f_m(x)\right]$$

约束方程

$$g_j(x) \le 0 \qquad j = 1,2,\Lambda\ , p$$

其中：$x=(x_1, x_2, \cdots, x_n)$ 为一个 n 维向量；$f_i(x)$为目标函数，i=1, 2, ⋯, m，$g_j(x)$为系统约束。

写成矩阵形式

目标函数

$$\min F(x)$$

约束方程

$$G(x) \le 0$$

当目标函数处于冲突状态时，不存在最优解使所有目标函数同时达到最优。此时，我们使用有效解，表示在不牺牲其他目标函数的前提下，不可能改进任何一个目标函数。也即：称 x^*为有效解，如果不存在 $x \in$ S 使得 $f_i(x) \geqslant f_i(x^*)$，$i$=1，2，…，$m$。且不等号至少对一个序号成立。一个有效解又称为非支配解、非劣解或者 Pareto 解。所有有效解构成的集合实际上

是一个有效前沿面。

如果决策者将 m 个目标函数集成在一起，构成一个实值偏好函数，则可以在相同的约束条件下，极大化偏好函数。这个模型称为妥协模型，其解称为妥协解。常见的妥协模型是通过对目标函数进行加权建立的。

第二种方法是极小化

$$\left(f_1(x), f_2(x), \Lambda, f_m(x)\right)$$

到一个理想的向量

$$\left(f_1^*, f_2^*, \Lambda, f_m^*\right)$$

的距离函数，其中 f_i^* 为第 i 个目标在不考虑其他目标时的最优解。

第三种是利用人机交互法去寻找妥协解。目前已得到各种人机交互法，如可行域削减法、加权向量空间削减法、准则锥削减法及线性搜索法。

在 Matlab 中，多目标规划问题使用一种由 Gembicki 提出的 Goal Attainment 法，可以将该方法归为距离最小化方法的一种。它将原问题化为以下问题求解

目标函数

$$\min_{x,\gamma} \gamma$$

约束方程

$$f_i(x) - w\gamma \le f_i^* \quad i = 1, \Lambda\ m$$

其中：$F=(f_1^*, f_2^*, ..., f_m^*)$为用户设计的与目标函数相应的目标值向量。$w=(w_1,w_2,\ldots,w_m)$为权值系数向量，它用于控制相对应的目标函数与用户定义的目标函数值的接近程度。γ 为一个松弛因子标量。当约束为二维函数时有

目标函数

$$\min_{x,\gamma} \gamma$$

约束方程

$$F_1(x) - w\gamma \le F_1^*$$
$$F_1(x) - w\gamma \le F_2^*$$

其中：点 $P(F_1^*, F_2^*)$ 为用户设计的目标点，W 权值向量定义了从 P 点到可行域的搜索方向。该问题的最优解为点(F_1, F_{2s})。

对上述问题的解法，优化工具箱采用 SQP 方法。SQP 方法详见 4.4.3 节。

在优化工具箱中，attgoal 函数用于求解上述优化问题。要求解多目标规划，用户必须首先对原问题进行转化。

- `attgoal`

  功能：求解多目标优化问题。

格式：x = attgoal('fun', x0, goal, w)

x = attgoal('fun', x0, goal, w, options)

x = attgoal('fun', x0, goal, w, options, vlb, vub)

x = attgoal('fun', x0, goal, w, options, vlb, vub, 'grad')

x = attgoal('fun', x0, goal, w, options, vlb, vub, 'grad', p1, p2, ...)

[x, options] = attgoal('fun', x0, ...)

说明：该函数求解下述优化问题。

目标函数

$$\min_{x,\gamma} \gamma$$

约束方程

$$F(x) - w\gamma \le goal$$

其中：$F(x)$ 为多目标规划中的目标函数向量，*goal* 为用户设计的目标函数值向量，*w* 为权值向量，它定义了搜索方向。γ 为一个松弛因子标量。

x = attgoal('fun', x0, goal, w)求解上述优化问题。约束方程定义在 M 文件中，文件名为 fun.m。初始解向量为 x0。目标函数值向量为 goal，权值向量为 w。

x = attgoal('fun', x0, goal, w, options)设置可选参数的值而不是采用缺省值。可选参数在 options 向量中设置。options 的详细用法见 4.2.3 节。

x = attgoal('fun', x0, goal, w, options, vlb, vub)设置解向量的上下界，即解向量必须满足 vlb<=x<=vub。

x = attgoal('fun', x0, goal, w, options, vlb, vub, 'grad')使用由函数 grad 计算的梯度信息。函数 grad 在 M 文件 grad.m 中定义。缺省时，梯度将通过有限差分方法计算导数来获得。

x = attgoal('fun', x0, goal, w, options, vlb, vub, 'grad', p1, p2, ...)传递附加的参数 p1, p2 等。附加参数的详细用法见 4.2.3 节。

[x, options] = attgoal('fun', x0, ...)同时返回在计算中使用的参数向量 options。

其中

x_0 为初始解向量。

fun 为包含多目标规划问题的目标函数的文件名字符串。如果目标函数要求尽可能地接近相应的 goal 向量的元素，则应将这几个元素置于目标函数的前几个元素，并用 options(15)设置它的个数。对 fun 函数也可以用字符串表达式来描述，例如：`x = attgoal(  in(x.*x)? x0,goal,w)`。goal 为用户设计的与目标函数相应的目标值向量。

w 为权值系数向量，它用于控制相对应的目标函数与用户定义的目标函数值的接近程度。当某一元素为正时，则算法将试图使目标函数的值小于用户定义的值；当某一元素为负时，则算法将试图使目标函数的值大于用户定义的值。要使所有的目标函数与用户定义的目标函数值具有同样的接近度，应设置‘w=abs(goal)’　。

vlb 为下界约束，vub 为上界约束。一般地，vlb 和 vub 具有和 x 同样的大小。如果 vlb 有 n 个元素且比 x 的元素少，则只有 x 的前 n 个元素具有下界约束。vub 变量也遵守同样的原则。

options 为参数控制向量。在 options 的 18 个元素中，attgaol 函数使用的输入参数包括 1、2、3、4、7、9、13、14、15、16、17，constr 函数使用的输出参数包括 8、10、11、18。其中

options(1)控制显示。设置为 1 将给出中间结果的列表显示。

options(2)控制 x 的精度。

options(3) 控制目标函数 f 的精度。

options(4)控制对约束的越限程度。

当 options(2)、options(3)、options(4)参数全部满足后，算法将结束。

options(7)、options(8)用于算法控制。

对 options 详细讨论请见 4.2.3 节。

grad 为包含梯度函数的函数名字符串，该函数具有 df = grad(x)格式。其中 df 为向量，它是目标函数 f 对于向量 x 的偏微分。

p1,p2,…为传递给 fun 和 grid 的附加参数。附加参数的详细用法见 4.2.3 节。

举例：(1) 控制系统输出反馈器设计。

设如下线性系统

$$\dot{x} = Ax + Bu$$

$$Y = Cx$$

其中

$$A = \begin{bmatrix} -0.5 & 0 & 0 \\ 0 & -2 & 10 \\ 0 & 1 & -2 \end{bmatrix} \qquad B = \begin{bmatrix} 1 & 0 \\ 2 & 2 \\ 0 & 1 \end{bmatrix} \qquad C = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

今要求设计输出反馈控制器 K，使闭环系统

$$\dot{x} = \left(A + BKC\right)x + Bu$$

的极点为[−5，−3，−1]，并要求

$$-4 \le K_{ij} \le 4\left(i, j = 1,2\right)$$

分析：上述问题即是要求解矩阵 K，使矩阵(A+BKC)的极点为[−5，−3，−1]，即为一个多目标规划问题。

求解

第一步：编写 M 文件

```
function F = fun(K,A,B,C)
F = sort(eig(A+B*K*C));
```

第二步： 参数输入

```
A = [-0.5 0 0; 0 -2 10; 0 1 -2];
```

```
B = [1 0; 2 2; 0 1];
C = [1 0 0; 0 0 1];
K = zeros(2,2)          % 初始化控制器矩阵
goal = [-5 -3 -1];      % 设置极点
w = abs(goal)           % 设置权值向量
vlb = -4*ones(size(K)); % 设置控制器下界
vub = 4*ones(size(K));  % 设置控制器上界
options = 1;            % 设置显示参数
```

第三步：求解

```
[K,options]
=attgoal(  un? K,goal,w,options,vlb,vub,[ ],A,B,C)
f-COUNT     MAX{g}         STEP  Procedures
    6      2.16228          1
   12      1.40287          1
   18     0.761498          1   Hessian modified
   24     0.704799          1   Hessian modified
   30      0.69568          1
   36     0.676085          1
   42     0.670218          1   Hessian modified
   48     0.669921          1   Hessian modified twice
   54      0.49996          1   Hessian modified
   61      1.02212        0.5   Hessian modified twice
   67      2.13437          1
   73     0.474145          1   Hessian modified
   81    -0.335485       0.25
   87    -0.340722          1   Hessian modified
   93    -0.342061          1   Hessian modified
   99    -0.342637          1   Hessian modified
  105    -0.369842          1
  111    -0.385115          1
  117    -0.386258          1   Hessian modified twice
  118     -0.38626          1   Hessian modified twice
Optimization Converged Successfully
Active Constraints:
1
2
K =
4.0000   0.2564
4.0000  -4.0000
```

```
options =
Columns 1 through 7
1.0000   0.0001   0.0001   0.0000      0      0      0
Columns 8 through 14
0.3863   0  118.0000   20.0000   20.0000    0   500.0000
Columns 15 through 18
0    0.0000    0.1000    1.0000
```

(2) 仍以上例问题不变，今要求每个极点都尽可能接近要求值(设置options(15)=3)。

```
options(15) = 3;
[K,options]=attgoal(  un? K,goal,w,options,vlb,vub,...
[ ],A,B,C)
After 37 function evaluations the solution is
K =
2.4294 -0.4891
3.9999 -2.0706
fun(K,A,B,C)
ans =
5.0000
3.0000
1.0000
The attainment factor is
options(8)
ans =
1.0859e-20
```

4.8 最小二乘最优

4.8.1 问题概述

最小二乘最优问题的一般描述

目标函数

$$\min_{x} \tfrac{1}{2}\|F(x)\|_2^2$$

约束方程

$$G(x) \le 0$$

其中：$F(x)$,G(x)函数向量。

如果目标函数具有 $F(x)=Ax-b$ 形式，则为线性最小二乘最优问题。其中 A 为常数矩阵，b 为向量。

并且如果

1. 无约束方程，则就是一个无约束线性最小二乘最优问题，可用优化工具箱中的 ‘\’ 函数来进行求解。
2. 约束方程具有 x≥0 的形式，则为一个非负约束最小二乘最优问题，可用函数 nnls 进行求解。
3. 约束方程具有 cx≤d 的形式，则为一个线性约束最小二乘最优问题，可用函数 conls 进行求解。

如果目标函数具有 $F(x)=[f_1(x)\ f_2(x)\ ?\ \ f_m(x)]'$ ，则目标函数可改写为

$$\min_x \tfrac{1}{2}\|F(x)\|_2^2 = \min_x \tfrac{1}{2}\sum_i f_i(x)^2$$

该问题就是一个非线性最小二乘最优问题(无约束条件)，可用函数 leastsq 进行求解。

如果目标函数具有如下形式(无约束方程)

$$\min_x \tfrac{1}{2}\|F(x, xdata) - ydata\|_2^2 = \tfrac{1}{2}\sum_i \left(F(x, xdata_i) - ydata_i\right)^2$$

其中：$xdata$ 为输入向量，$ydata$ 为输出向量，现仅已知输入与输出的函数关系为 $ydata=F(x,xdata)$，但不知系数向量 x，则构成最小二乘拟合问题。目标函数的值称为拟合方差，可用 curvefit 函数进行求解。

- **非线性最小二乘最优问题的优化算法**

如上所述，非线性最小二乘最优问题可表示为

$$\min_x \tfrac{1}{2}\|F(x)\|_2^2 = \min_x \tfrac{1}{2}\sum_i f_i(x)^2$$

对上述问题可以使用一般的无约束非线性规划问题的解法来解决，但注意到 LS 方程的梯度向量和 Hessian 矩阵的特殊性，则可提高求解的迭代效率。设 $F(x)$的梯度向量为▽$F(x)$，其 Jacobian 矩阵为 $J(x)$，其 Hessian 矩阵为 $H(x)$。则有

$$\nabla f(x) = 2J(x)F(x)$$

$$H(x) = 2J(x)^T J(x) + 2Q(x)$$

$$Q(x) = \sum_{i=1}^{m} f_i(x)H_i(x)$$

即当 x 趋近于最优解时，随着 $F(x)$趋近于零，$Q(x)$也趋近于零。因此可以考虑采用 Gauss-Newton 方向作为每次迭代的搜索方向。

Gauss-Newton 法

在 Gauss-Newton 法中，每次迭代时采用的搜索方向 d_k 通过求解以下线性最小二乘问题

获得

$$\min_{x \in R^n} \| J(x_k) d_k - F(x_k) \|_2^2$$

Levenberg-Marquardt 法

在 Levenberg-Marquardt 法中，搜索方向 d_k 通过求解以下线性方程组获得

$$\left(J(x_k)^T J(x_k) + \lambda_k I \right) d_k = -J(x_k) F(x_k)$$

这里，标量 λ_k 可以控制 d_k 的大小和方向。当 $\lambda_k=0$ 时，d_k 的方向与 Gauss-Newton 法的搜索方向一致。

4.8.2 nnls 函数——非负线性最小二乘求解

非负线性最小二乘问题的一般形式

目标函数

$$\min_{x} \tfrac{1}{2} \| Ax - b \|_2^2$$

约束条件

$$x \geq 0$$

其中：矩阵 A 和向量 b 为目标函数的系数。

在优化工具箱中，nnls 函数用于求解非负线性最小二乘问题。

- nnls

功能：求解非负最小二乘问题。

格式：nnls(A,b)

nnls(A,b,tol)

[x,w] = nnls(A,b)

[x,w] = nnls(A,b,tol)

说明：x=nnls(A,b)求解上述非负最小二乘问题，返回解向量 x。

x=nnls(A,b,tol)定义 x 的容许误差，而不是使用缺省值。x 的容许误差用来决定 x 向量的元素是否已经小于 0。缺省的容许误差为：tol = max(size(A)) * norm(A,1) * eps。

[x,w] = nnls(A,b)同时返回向量 w，x 和 w 的关系为

w(i) <= 0, (i | x(i) = 0)

w(i) = 0, (i | x(i) > 0)

举例：一个最小二乘问题的无约束与非负约束解法的比较。

第一步：输入系数

```
a =
```

```
0.0372 0.2869
0.6861 0.7071
0.6233 0.6245
0.6344 0.6170
b =
0.8587
0.1781
0.0747
0.8405
```

第二步：求解

```
[a\b, nnls(a,b)] = -2.5625  0
3.1106  0.6929
[norm(a*(a\b)-b), norm(a*nnls(a,b)-b)] = 0.6677  0.9119
```

注意：1. 当问题为无约束线性最小二乘问题时，使用 Matlab 下的 ’算子即可以解决。关于‘\’的使用详见附录。

2. 非负最小二乘问题是线性约束最小二乘问题的一类子问题，因此，当矩阵 A 的行数比列数多时，调用

```
[x,w] = nnls(A,b)
```

等价于

```
[m,n] = size(A);
[x,l] = conls(A,b,-eye(n,n),zeros(n,1));
w = -1;
```

3. 一般地，对于非负最小二乘问题，nnls 函数更有效，但当矩阵行数大于 20 时，conls 有可能比 nnls 要快。

4.8.3　conls 函数——约束线性最小二乘求解

线性约束最小二乘问题的一般描述

目标函数

$$\min \tfrac{1}{2}\|Ax-b\|_2^2$$

约束条件

$$Cx \le d$$

其中：A,C 为矩阵，b,d,x 为向量。

在优化工具箱中，conls 函数用于求解线性约束最小二乘问题。

- conls

功能：线性约束最小二乘问题求解。

格式：x = conls(A,b,C,d)

x = conls(A,b,C,d,vlb)

x = conls(A,b,C,d,vlb,vub)

x = conls(A,b,C,d,vlb,vub,x0)

x = conls(A,b,C,d,vlb,vub,x0,neqcstr)

x = conls(A,b,C,d,vlb,vub,x0,neqcstr,display)

[x,lambda,how] = conls(A,b,C,d, ...)

说明：x = conls(A,b,C,d)求解在约束 c*x<=d 下方程 A*x=b 的最小二乘解。其解向量由变量 x 返回。

x = conls(A,b,C,d,vlb,vub) 设置解向量的上下界，即解向量必须满足 vlb<=x<=vub。

x = conls(A,b,C,d,vlb,vub,x0)设置初始解向量 x_0。

x = conls(A,b,C,d,vlb,vub,x0,neqcstr)设置在约束中的等式约束的个数。等式约束必须位于约束方程的前面几个。

x = conls(A,b,C,d,vlb,vub,x0,neqcstr,display)设置警告信息的显示。

[x,lambda] = conls(A,b,C,d, ...)同时返回拉格朗日(Lagrange)乘子 lambda。

[x,lambda,how] = conls(A,b,C,d, ...) 同时返回在最后一次调用过程中的错误状态。该错误状态字符串由变量 how 返回。

其中

A,b 为线性系统的系数。

C,d 为线性约束的系数。等式约束的系数必须位于矩阵 C 的前几行和向量 d 的前几个元素。

vlb 为下界约束，vub 为上界约束。一般地，vlb 和 vub 具有和 x 同样的大小。如果 vlb 由 n 个元素且比 x 的元素少，则只有 x 的前 n 个元素具有下界约束。vub 变量也遵守同样的原则。

x_0 为初始解向量。缺省时，x_0=zeros(size(x))。设置初始解向量可以使运算快速收敛。对于一个病态问题，好的初始解向量可以得到一个更好的解。

neqcstr 为等式约束的个数。

display 为控制警告信息显示的标志位。缺省时 display=0，即显示警告信息。如果 display=−1，则不显示警告信息。

lambda 为解的拉格朗日(Lagrange)乘子向量。拉格朗日乘子向量的长度等于 length(b)+length(vlb)+length(vub)，且元素顺序对应为 A、vlb、vub。

how 为求解过程中的错误状态指示字符串。如果 how='infeasible'，则该问题无可行解；如果 how='unbounded'，则该问题有无界解；如果 how='dependent'，则该问题的约束中有相关的等式约束，在求解过程中相关约束已经去除；如果 how='ok'，则该问题正常结束。

和其他优化工具箱中的函数一样。空的调用系数将导致调用过程中使用缺省值。例如：`conls(A,b,C,d,[ ],[ ],[ ],length(b))`表明该问题为一个等式约束问题，无上下界约束并使用缺省的初始解向量。

举例：求解如下系统的最小二乘解。

系统

Ax=b

约束

Cx≤b；vlb≤x≤vub

第一步：输入系统系数

```
A =
0.9501 0.7620 0.6153 0.4057
0.2311 0.4564 0.7919 0.9354
0.6068 0.0185 0.9218 0.9169
0.4859 0.8214 0.7382 0.4102
0.8912 0.4447 0.1762 0.8936
b =
0.0578
0.3528
0.8131
0.0098
0.1388
C =
0.2027 0.2721 0.7467 0.4659
0.1987 0.1988 0.4450 0.4186
0.6037 0.0152 0.9318 0.8462
d =
0.5251
0.2026
0.6721
vlb = 0.1*ones(4,1);
vub = 2*ones(4,1);
```

第二步：求解

```
[x,lambda] = conls(A,b,C,d,vlb,vub)
```

解为

```
x =
0.1000
0.1000
0.2152
0.3502
lambda =
0
```

```
0.2392
0
0.0409
0.2784
0
0
0
0
0
0
```

可见，拉格朗日乘子(lambda)的前三个元素与非等式约束相关，lambda 的非零元素表示对应的约束为有效约束。在本问题中，第二个线性不等式约束和前两个下界约束为有效约束。

注意：1. 当问题无可行解时将给出以下信息：

```
Warning: The constraints are overly stringent;
there is no feasible solution.
```

即约束条件过于严格。此时，conls 将给出一个对约束的破坏影响最小的解。

2. 当等式约束之间不相容时将给出以下信息：

```
Warning: The equality constraints are overly stringent;
there is no feasible solution.
```

3. 当 Hessian 矩阵 H 半负定时，问题将具有无界解，此时将给出以下信息：

```
Warning: The solution is unbounded and at infinity;
the constraints are not restrictive enough.
```

4. 对于无约束问题，应采用算子 '\'。

4.8.4 leastsq 函数——非线性最小二乘求解

非线性最小二乘问题的一般描述

目标函数

$$\min_{x} f(x) = f_1(x)^2 + f_2(x)^2 + \ldots + f_m(x)^2$$

设

$$F(x) = \begin{bmatrix} f_1(x) \\ f_2(x) \\ \vdots \\ f_m(x) \end{bmatrix}$$

则目标函数可表述为

$$\min_{x} \frac{1}{2}\|F(x)\|_2^2 = \frac{1}{2}\sum_{i} f_i(x)^2$$

其中：x 为向量，L 为常数标量，$F(x)$为函数向量。

在优化工具箱中，leastsq 函数用于求解非线性最小二乘问题。该函数采用 Levenberg-Marquardt 方法或者 Gauss-Newton 方法，可以通过设置 options 向量来进行选择。

- `leastsq`

功能：求解非线性最小二乘(非线性数据拟合)问题。

格式：x = leastsq('fun',x0)

x = leastsq('fun',x0,options

x = leastsq('fun',x0,options, 'grad')

x = leastsq('fun',x0,options, 'grad',p1,p2, ...)

[x,options] = leastsq('fun',x0, ...)

[x,options,funval] = leastsq('fun',x0, ...)

[x,options,funval,jacob] = leastsq('fun',x0, ...)

说明：x = leastsq('fun',x0) 求解非线性最小二乘问题，返回解向量 x。目标函数 fun 定义在 M 文件 fun.m 中，并置初始解向量为 x0。

x = leastsq('fun',x0,options) 设置可选参数的值而不是采用缺省值。可选参数在 options 向量中设置。options 的详细用法见 4.2.3。

x = leastsq('fun',x0,options, 'grad') 使用由函数 grad 计算的梯度信息。函数 grad 在 M 文件 grad.m 中定义。缺省时，梯度将通过有限差分方法计算目标函数的导数来获得。

x = leastsq('fun',x0,options, 'grad',p1,p2, ...) 传递附加的参数 p1, p2 等。附加参数的详细用法见 4.2.3。

[x,options] = leastsq('fun',x0, ...) 同时返回在计算中使用的一些参数。例如 options[10]包含计算过程中目标函数的计算次数。

[x,options,funval] = leastsq('fun',x0, ...) 同时将在解 x 处的函数值 fun(x) 返回给 funval。

[x,options,funval,jacob] = leastsq('fun',x0, ...) 同时返回在解 x 处函数 fun 的 Jacobian 矩阵。其中

fun 为包含目标函数的函数名字符串。fun 函数返回 1 个向量参数：目标函数的函数值 f，例如 `f = fun(x,xdata)`。也可以用字符串表达式达到同样的目的，例如 `x = fsolve(  in(x.*x)? x0)`，grad 为包含梯度函数的函数名字符串，该函数具有 df = grad(x,xdata)格式。其中 df 为矩阵，它是函数 F 对于向量 x 的偏微分。

options 为参数控制向量。在 options 的 18 个元素中，leastsq 函数使用的输入参数包括 1、2、3、5、7、9、14、16、17，leastsq 函数使用的输出参数包括 8、10、11、18。其中

options(1)控制显示，设置为 1 将给出中间结果的列表显示。

options(2)控制 x 的精度。

options(3)控制目标函数 f 的精度。

options(5)和 options(7)含义见后面算法部分。

当 options(2)、options(3)参数全部满足后，算法将结束。

对 options 详细讨论请见 4.2.3 节。

x0 为初始解向量。

p1,p2,…为传递给 fun 和 grid 的附加参数。关于附加参数的用法见 4.2.3 节。

funval 为在解 x 处的函数值 fun(x)。

jacob 为在解 x 处函数的 Jacobian 矩阵。

举例：求下述非线性最小二乘问题：

$$\sum_{k=1}^{10}\left(2+2k-e^{kx_1}-e^{kx_2}\right)^2$$

初始解向量为 x=[0.3 0.4]。

由于 leastsq 函数要求传递的函数为向量形式且不具有平方和形式，因此对函数做以下变换：

$$F_k(x)=2+2k-e^{kx_1}-e^{kx_2} \qquad k=1,2,\cdots,10。$$

第一步：编写 M 文件

```
function f = fun(x)
k = [1:10];
f = 2 + 2*k-exp(k*x(1))-exp(k*x(2));
```

第二步：求解

```
x0 = [0.3 0.4]
x = leastsq(  un? x0)
```

经过 41 次运算，得到以下结果

```
x =
0.25783  0.25783
sum(fun(x).*fun(x)) %求目标函数值
ans =
124.3622
```

算法：对 leastsq 函数算法的选择通过 options(5) 参数实现，缺省地 options(5)=0，此时使用 Levenberg-Marquardt 方法；如果设置 options(6)=1，则采用 Gauss-Newton 方法。当 $\|F(x)\|_2^2$ 比较小时这种方法较快速。

线性搜索算法的选择通过 options(7)参数实现。缺省地 options(7)=0，使用一种二次和三次多项式插值的混合算法；如果 options(7)=1，使用三次多项式插值算法，该算法需要较少的函数计算和较多的梯度计算。

注意：leastsq 函数只处理连续函数，且有可能给出局部最小值。

4.8.5　curvefit 函数——非线性数据拟合

最小二乘非线性数据拟合问题的一般描述

设存在输入向量 *xdata* 和输出向量 *ydata*，现仅已知输入与输出的函数关系为 $ydata=F(x,xdata)$，但不知系数向量 x。今进行曲线拟合，求取 x 使得下式成立

$$\min_{X} \frac{1}{2}\left\|F(x,xdata)-ydata\right\|_2^2=\frac{1}{2}\sum_i\left(F(x,xdata_i)-ydata_i\right)^2$$

在优化工具箱中，curvefit 函数用于求解最小二乘非线性数据拟合问题。该函数采用 Levenberg-Marquardt 方法或者 Gauss-Newton 方法，可以通过设置 options 向量来进行选择。

- curvefit

功能：求解最小二乘非线性数据拟合问题。

格式：
```
x = curvefit('fun',x0,xdata,ydata)
x = curvefit('fun',x0,xdata,ydata,options)
x = curvefit('fun',x0,xdata,ydata,options, 'grad')
x = curvefit('fun',x0,xdata,ydata,options, 'grad',p1,p2, ... )
[x,options] = curvefit('fun',x0,xdata,ydata ... )
[x,options,funval] = curvefit('fun',x0,xdata,ydata ... )
[x,options,funval,jacob] = curvefit('fun',x0,xdata,ydata ... )
```

说明：在 curvefit 函数调用中，必须有一个用户定义的可计算向量值的拟合函数 F(x,xdata)。而且返回向量必须和 ydata 同样大小。

x = curvefit('fun',x0,xdata,ydata) 求解最小二乘非线性数据拟合问题。拟合函数定义在 M 文件中，文件名为 fun.m。初始向量为 x0。

x = curvefit('fun',x0,xdata,ydata,options) 设置可选参数的值而不是采用缺省值。可选参数在 options 向量中设置。options 的详细用法见 4.2.3 节。

x = curvefit('fun',x0,xdata,ydata,options, 'grad') 使用由函数 grad 计算的梯度信息。函数 grad 在 M 文件 grad.m 中定义。缺省时，梯度将通过有限差分方法计算拟合函数的导数来获得。

x = curvefit('fun',x0,xdata,ydata,options, 'grad',p1,p2, ...) 传递附加的参数 p1，p2 等。附加参数的详细用法见 4.2.3 节。

[x,options] = curvefit('fun',x0,xdata,ydata, ...) 同时返回在计算中使用的一些参数。例如 options[10]包含计算过程中目标函数的计算次数。

[x,options,funval] = curvefit('fun',x0,xdata,ydata, ...)同时返回在解 x 处的函数值 fun(x)。

[x,options,funval,jacob] = curvefit('fun',x0,xdata,ydata, ...) 同时返回在解 x 处函数 fun 的 Jacobian 矩阵。

其中

fun 为包含拟合函数的函数名字符串。fun 函数返回 1 个参数：拟合函数的函

数值 f，例如 `f = fun(x,xdata)`。

grid 为包含梯度函数的函数名字符串，该函数具有 df = grad(x,xdata)格式。其中 df 为向量，它是拟合函数 f 对于向量 x 的偏微分。

options 为参数控制向量。在 options 的 18 个元素中，curvefit 函数使用的输入参数包括 1、2、3、5、7、9、14、16、17，curvefit 函数使用的输出参数包括 8、10、11、18。其中

options(1)控制显示。设置为 1 将给出中间结果的列表显示。

options(2)控制 x 的精度。

options(3)控制拟合函数 f 的精度。

options(5)和 options(7)含义见后面算法部分。

当 options(2)、options(3)参数全部满足后，算法将结束。

对 options 详细讨论请见 4.2.3 节。

x0 为初始解向量。

p1,p2,…为传递给 fun 和 grid 的附加参数。关于附加参数的用法见 4.2.3 节。

funval 为在解 x 处的函数值 fun(x)。

jacob 为在解 x 处函数 fun 的 Jacobian 矩阵。

举例：求解如下最小二乘非线性数据拟合问题。

已知输入向量 *xdata*；输出向量 *ydata*；且长度均为 *n*。

拟合函数

$$ydata(i)=x(1)+x(2)*e^{(xdata(i)+x(3))}$$

即目标函数为

$$\min_x \frac{1}{2}\sum_i \left(F\left(x, xdata_i\right)-ydata_i\right)^2$$

其中

$$F(x,xdata)=x(1)+x(2)*e^{(xdata(i)+x(3))}$$

初始解向量：*x*=[0.3 0.4 0.1]

第一步：编写 M 文件

```
function f = fun(x,xdata)
f = x(1)+x(2)*exp(xdata + x(3)); %注意 f 为向量
```

第二步：求解。假设 xdata、ydata 已存在

```
x0 = [0.3 0.4 0.1]
x = curvefit(  un? x0,xdata,ydata)
x =
0.25783 0.25783
sum(fun(x,xdata).*fun(x,xdata))
ans =
124.3622
```

算法：对算法的控制通过 options(5)参数实现。缺省 options(5)=0，使用 Levenberg-Marquardt 算法；若 options(5)=1，则使用 Gauss-Newton 方法。

对线性搜索算法的控制通过 options(7)参数实现。缺省时 options(7)=0，使用一种二次和三次多项式插值的混合算法；如果 options(7)=1，使用三次多项式插值算法，该算法需要较少的函数计算和较多的梯度计算。

4.9　方 程 求 解

优化工具箱同时提供了三个方程求解的函数。其中，'\' 算子可用于求解线性方程组 $Ax=b$：当矩阵 A 为 n 阶方阵时，采用高斯消元法进行求解；如果 A 不为方阵，则采用数值方法计算方程最小二乘意义上的解。fzero 采用数值解法求解非线性方程。fsolve 函数则采用非线性最小二乘算法求解非线性方程组。以下给出 fsolve 和 fzero 函数的使用说明。

1　fzero

功能：求解单变量函数。

格式：z = fzero('fun',x0)

z = fzero('fun',x0,tol)

z = fzero('fun',x0,tol,trace)

z = fzero('fun',x0,tol,trace,P1,P2,...)

说明：z = fzero('fun',x0) 单变量函数 fun 求解，返回函数的解 z。方程 fun 定义在 M 文件 fun.m 中，并置初始搜索点为 x0。

z = fzero('fun',x0)设置解的搜索区域 x0。其中 x0 为一个长度为 2 的向量，而且 f(x(1))的符号和 f(x(2))的符号不同，调用时如果符号相同则报错。

z = fzero('fun',x0,tol)设置解的精度。返回的函数值的相对误差必须在变量 tol 给定的范围以内。

z = fzero('fun',x0,tol,trace)设置显示每次的迭代信息。如果变量 trace 不等于零，则显示每次的迭代信息。

z = fzero('fun',x0,tol,trace,P1,P2,...) 传递附加的参数 p1, p2 等。附加参数的详细用法见 4.2.3 节。其中

fun 为包含任意单变量函数的文件名字符串。

x0 为函数零点处 x 坐标的初始估计或者零点处 x 坐标的所在范围。

tol 为相对容许误差。缺省时 tol=eps。

trace 为中间结果显示控制。如果 trace=0 则不显示每次的迭代信息；否则就显示。

p1,p2,…为传递给 fun 函数的附加参数。附加参数的详细用法见 4.2.3 节。

举例：对下述函数求解：

$$f(x)=x^3-2x-5$$

第一步：编写 M 文件

```
function y = f(x)
y = x.^3-2*x-5;
```

第二步：求解

```
z = fzero(  ? 2)
z =
2.0946
```

注意该函数为多项式，实际的零点有 3 个，其余两个为一对共轭复零点。由于 fzero 不能求复数零点，因此，任意的初始估计对将给出同样的结果。

注意：fzero 函数认为零点是函数穿过 x 轴的点，对于与 x 轴相切的点，该函数将不能计算出来。

2 fsolve

功能：非线性方程求解。

格式：x = fsolve('fun',x0)

x = fsolve('fun',x0,options)

x = fsolve('fun',x0,options, 'grad')

x = fsolve('fun',x0,options, 'grad',p1,p2, ...)

[x,options] = fsolve(un? x0, ...)

说明：非线性方程的一般描述

$$F(x)=0$$

其中：x 为向量，$F(x)$为一个函数向量。

x = fsolve('fun',x0) 非线性方程 fun 求根，返回解向量 x。方程 fun 定义在 M 文件 fun.m 中，并置初始解向量为 x0。

x = fsolve('fun',x0,options) 设置可选参数的值而不是采用缺省值。可选参数在 options 向量中设置。options 的详细用法见 4.2.3。

x = fsolve('fun',x0,options, 'grad') 使用由函数 grad 计算的梯度信息。函数 grad 在 M 文件 grad.m 中定义。缺省地，梯度将通过有限差分方法计算目标函数的导数来获得。

x = fsolve('fun',x0,options, 'grad',p1,p2, ...) 传递附加的参数 p1，p2 等。附加参数的详细用法见 4.2.3 节。

[x,options] = fsolve('fun',x0, ...) 同时返回在计算中使用的一些参数。例如 options[10]包含计算过程中目标函数的计算次数。其中

x_0 为初始解向量。

fun 为包含目标函数的文件名字符串。fun 函数返回 1 个参数：目标函数的函数值 f，f 为向量，并具有 [f] = fun(x) 格式。也可以用字符串表达式达到同样的目的，例如 `x = fsolve(  in(x.*x)? x0)`。

options 为参数控制向量。在 options 的 18 个元素中，fsolve 函数使用的输入参数包括 1、2、3、5、7、9、14、16、17，fsolve 函数使用的输出参数包括 8、10、11、18。其中

options(1)控制显示。设置为 1 将给出中间结果的列表显示。

options(2)控制 x 的精度。

options(3) 控制目标函数 f 的精度。

当 options(2)、options(3)参数全部满足后，算法将结束。

对 options(5)、options(7)的讨论见后面算法部分。

对 options 详细讨论请见 4.2.3 节。

grid 为包含梯度函数的函数名字符串，该函数具有 df = grad(x)格式。其中 df 为矩阵，它是目标函数 f 对于向量 x 的偏微分，也即函数向量 f 的 Jacobian 矩阵的转置。

p1,p2,…为传递给 fun 和 grid 的附加参数。附加参数的详细用法见 4.2.3 节。

举例：(1)　求下述系统的根。

$$2x_1 - x_2 = e^{-x_1}$$
$$-x_1 + 2x_2 = e^{-x_2}$$

即解下述方程

$$2x_1 - x_2 - e^{-x_1} = 0$$
$$-x_1 + 2x_2 - e^{-x_2} = 0$$

并设初始解向量为 x0=[−5 −5]。

第一步：编写 M 文件

```
function F = fun(x)
F = [2*x(1) -x(2) -exp(-x(1));

          x(1) + 2*x(2) -exp(-x(2))];
```

第二步：求解

```
x0 = -5*ones(2,1);
options=foptions;
options(1)=1; %设置显示输出中间结果
x = fsolve(  un? x0,options)
f = fun(x)
```

经过 25 次迭代后，得到一个零点

```
f-COUNT RESID      STEP-SIZE   GRAD/SD
   3     47071.2        1        -9.41e+04
   8     966.828        1        -1.81e+03
   15    1.99465      3.85        5.6
   20  0.000632051   0.895      -0.0867
   25  1.39647e?5    0.998      -0.89e?9
```

解及解处的函数值为：

```
x =
0.5671
0.5671
```

```
f =
1.0e-07*0.2642 0.2642
```

(2) 求矩阵 x 使其满足方程

$$x*x*x=\begin{bmatrix}1 & 2\\3 & 4\end{bmatrix}$$

并设初始解向量为：x=[1,1;1,1]

第一步：编写 M 文件

```
function F = fun(x)
F = x*x*x   1,2;3,4];
```

第二步：求解

```
x0 = ones(2,2);
x = fsolve(  un? x0)
```

经过 44 次迭代后，解为：

```
x =
0.1291 0.8602
1.2903 1.1612
F = x*x*x-[1,2;3,4]
F =
1.0e-05 *
0.0350 0.1268
0.0721 -0.1293
sum(sum(F.*F))
ans =
3.9218e -12
```

算法：fsolve 函数使用非线性最小二乘算法。使用该算法的优点在于如果系统没有根或者由于不够精确而得不到系统的根，则该算法仍然可以返回一个解。如果系统的 Jacobian 矩阵奇异，则该算法可能收敛到一个点，但该点不是方程的根。

对 fsolve 函数算法的选择通过 options(5) 参数实现，缺省时 options(5)=0，此时使用 Gauss-Newton 方法；如果设置 options(5)=1，则采用 Levenberg-Marquardt 方法。

线性搜索算法的选择通过 options(7)参数实现。缺省时 options(7)=0，使用一种二次和三次多项式插值的混合算法；如果 options(7)=1，使用三次多项式插值算法，该算法需要较少的函数计算和较多的梯度计算。

注意：1. 方程必须是连续的。

2. 该函数可能收敛至一个非零点，此时可以试一试不同的初始解向量。

参 考 文 献

[1] Optimization Toolbox User's Guide, MathWorks, 1997
[2] 甘应爱、李维铮、钱颂迪等著，运筹学，清华大学出版社，1990
[3] 赵瑞安、吴方著，非线性最优化理论与方法，浙江科学技术出版社，1992
[4] 谢骞会，最优化理论与方法，国防科大出版社，1992

附录　Matlab 函数参考

由于正文中各章是以工具箱为主，对于 Matlab 标准环境下的函数及命令涉及较少，为完整及查阅方便起见，现将 Matlab 下的标准函数以附录给出。这里仅仅给出函数的功能描述，使用命令 `help fun`(fun 为函数名)可以获得函数联机帮助。

附录 1　常 用 命 令

附录 1.1　管理用命令

函　数　名	功 能 描 述	函　数　名	功 能 描 述
addpath	增加一条搜索路径	rmpath	删除一条搜索路径
demo	运行 Matlab 演示程序	type	列出.M 文件
doc	装入超文本文档	version	显示 Matlab 的版本号
help	启动联机帮助	what	列出当前目录下的有关文件
lasterr	显示最后一条错误信息	whatsnew	显示 Matlab 的新特性
lookfor	搜索关键词的帮助	which	找出函数与文件所在的目录
path	设置或查询 Matlab 路径		

附录 1.2　管理变量与工作空间用命令

函　数　名	功 能 描 述	函　数　名	功 能 描 述
clear	删除内存中的变量与函数	pack	整理工作空间内存
disp	显示矩阵或文本	save	将工作空间中的变量存盘
length	查询向量的维数	size	查询矩阵的维数
load	从文件中装入数据	who,whos	列出工作空间中的变量名

附录 1.3　文件与操作系统处理命令

函　数　名	功 能 描 述	函　数　名	功 能 描 述
cd	改变当前工作目录	edit	编辑.M 文件
delete	删除文件	matlabroot	获得 Matlab 的安装根目录
diary	将 Matlab 运行命令存盘	tempdir	获得系统的缓存目录
dir	列出当前目录的内容	tempname	获得一个缓存（temp）文件
!	执行操作系统命令		

附录 1.4　窗口控制命令

函　数　名	功 能 描 述	函　数　名	功 能 描 述
echo	显示文件中的 Matlab 中的命令	more	控制命令窗口的输出页面
format	设置输出格式		

附录 1.5　启动与退出命令

函　数　名	功 能 描 述	函　数　名	功 能 描 述
matlabrc	启动主程序	quit	退出 Matlab 环境
startup	Matlab 自启动文件		

附录 2　运算符号与特殊字符

附录 2.1　运算符号与特殊字符

函　数　名	功 能 描 述	函　数　名	功 能 描 述
+	加	?	续行标志
-	减	,	分行符(该行结果不显示)
*	矩阵乘	;	分行符(该行结果显示)
.*	向量乘	%	注释标志
^	矩阵乘方	!	操作系统命令提示符
.^	向量乘方	'	矩阵转置
kron	矩阵 kron 积	.'	向量转置
\	矩阵左除	=	赋值运算
/	矩阵右除	= =	关系运算之相等
.\	向量左除	< >	关系运算之不等
./	向量右除	<	关系运算之小于
:	向量生成或子阵提取	<=	关系运算之小于等于
()	下标运算或参数定义	>	关系运算之大于
[]	矩阵生成	>=	关系运算之大于等于
{}		&	逻辑运算之与
.	结构字段获取符	|	逻辑运算之或
.	点乘运算，常与其他运算符联合使用（如.\）	~	逻辑运算之非
xor	逻辑运算之异或		

附录 2.2　逻辑函数

函　数　名	功 能 描 述	函　数　名	功 能 描 述
all	测试向量中所有元素是否为真	is*（一类函数）	检测向量状态。其中*表示一个确定的函数(如 isinf)
any	测试向量中是否有真元素	*isa	检测对象是否为某一个类的对象
exist	检验变量或文件是否定义	logical	将数字量转化为逻辑量
find	查找非零元素的下标		

附录 3　语言结构与调试

附录 3.1　编程语言

函　数　名	功 能 描 述	函　数　名	功 能 描 述
builtin	执行 Matlab 内建的函数	global	定义全局变量
eval	执行 Matlab 语句构成的字符串	nargchk	函数输入输出参数个数检验
feval	执行字符串指定的文件	script	Matlab 语句及文件信息
function	Matlab 函数定义关键词		

附录 3.2　控制流程

函　数　名	功 能 描 述	函　数　名	功 能 描 述
break	中断循环执行的语句	if	条件转移语句
case	与 switch 结合实现多路转移	otherwise	多路转移中的缺省执行部分
else	与 if 一起使用的转移语句	return	返回调用函数
elseif	与 if 一起使用的转移语句	swith	与 case 结合实现多路转移
end	结束控制语句块	warning	显示警告信息
error	显示错误信息	while	循环语句
for	循环语句		

附录 3.3　交互输入

函　数　名	功 能 描 述	函　数　名	功 能 描 述
input	请求输入	menu	菜单生成
keyboard	启动键盘管理	pause	暂停执行

附录 3.4　面向对象编程

函　数　名	功 能 描 述	函　数　名	功 能 描 述
class	生成对象	isa	判断对象是否属于某一类
double	转换成双精度型	superiorto	建立类的层次关系
inferiorto	建立类的层次关系	unit8	转换成 8 字节的无符号整数
inline	建立一个内联对象		

附录 3.5　调试

函　数　名	功 能 描 述	函　数　名	功 能 描 述
dbclear	清除调试断点	dbstatus	列出所有断点情况
dbcont	调试继续执行	dbstep	单步执行
dbdown	改变局部工作空间内容	dbstop	设置调试断点
dbmex	启动对 Mex 文件的调试	dbtype	列出带命令行标号的.M 文件
dbquit	退出调试模式	dbup	改变局部工作空间内容
dbstack	列出函数调用关系		

附录 4　基本矩阵及矩阵处理

附录 4.1　基本矩阵

函　数　名	功 能 描 述	函　数　名	功 能 描 述
eye	产生单位阵	rand	产生随机分布矩阵
linspace	构造线性分布的向量	randn	产生正态分布矩阵
logspace	构造等对数分布的向量	zeros	产生零矩阵
ones	产生元素全部为 1 的矩阵	:	产生向量

附录 4.2　特殊向量与常量

函　数　名	功 能 描 述	函　数　名	功 能 描 述
ans	缺省的计算结果变量	NaN	非数值常量(常由 0/0 或 Inf/Inf)获得
computer	运行 Matlab 的机器类型	nargin	函数中参数输入个数
eps	精度容许误差	nargout	函数中输出变量个数
flops	浮点运算计数	pi	圆周率 π
i	复数单元	realmax	最大浮点数值
Inf	无穷大	realmin	最小浮点数值
inputname	输入参数名	varargin	函数中输入的可选参数
j	复数单元	varargout	函数中输出的可选参数

附录 4.3　时间与日期

函　数　名	功 能 描 述	函　数　名	功 能 描 述
calender	日历	eomday	计算月末
clock	时钟	etime	所用时间函数
cputime	所用的 CPU 时间	now	当前日期与时间
date	日期	tic	启动秒表计时器
datenum	日期(数字串格式)	toc	读取秒表计时器
datestr	日期(字符串格式)	weekday	星期函数
datevoc	日期(年月日分立格式)		

附录 4.4　矩阵处理

函　数　名	功 能 描 述	函　数　名	功 能 描 述
cat	向量连接	reshape	改变矩阵行列个数
diag	建立对角矩阵或获取对角向量	rot90	将矩阵旋转 90 度
f liplr	按左右方向翻转矩阵元素	tril	取矩阵的下三角部分
flipud	按上下方向翻转矩阵元素	triu	取矩阵的上三角部分
repmat	复制并排列矩阵函数		

附录 5 特 殊 矩 阵

函 数 名	功 能 描 述	函 数 名	功 能 描 述
compan	生成伴随矩阵	invhilb	生成逆 hilbert 矩阵
gallery	生成一些小的测试矩阵	magic	生成 magic 矩阵
hadamard	生成 hadamard 矩阵	pascal	生成 pascal 矩阵
hankel	生成 hankel 矩阵	toeplitz	生成 toeplitz 矩阵
hilb	生成 hilbert 矩阵	wilkinson	生成 wilkinson 特征值测试矩阵

附录 6 数 学 函 数

附录 6.1 三角函数

函 数 名	功 能 描 述	函 数 名	功 能 描 述
sin/asin	正弦/反正弦函数	sec/asec	正割/反正割函数
sinh/asinh	双曲正弦/反双曲正弦函数	sech/asech	双曲正割/反双曲正割函数
cos/acos	余弦/反余弦函数	csc/acsc	余割/反余割函数
cosh/acosh	双曲余弦/反双曲余弦函数	csch/acsch	双曲余割/反双曲余割函数
tan/atan	正切/反正切函数	cot/acot	余切/反余切函数
tanh/atanh	双曲正切/反双曲正切函数	coth/acoth	双曲余切/反双曲余切函数
atan2	四个向限内反正切函数		

附录 6.2 指数函数

函 数 名	功 能 描 述	函 数 名	功 能 描 述
exp	指数函数	log10	常用对数函数
log	自然对数函数	sqrt	平方根函数

附录 6.3 复数函数

函 数 名	功 能 描 述	函 数 名	功 能 描 述
abs	绝对值函数	imag	求虚部函数
angle	角相位函数	real	求实部函数
conj	共轭复数函数		

附录 6.4 数值处理

函 数 名	功 能 描 述	函 数 名	功 能 描 述
fix	沿零方向取整	round	截取到最近的整数
floor	沿−∞方向取整	rem	求除法的余数
ceil	沿+∞方向取整	sign	符号函数

附录 6.5　其他特殊数学函数

函　数　名	功能描述	函　数　名	功能描述
airy	airy 函数	erfcx	比例互补误差函数
besselh	bessel 函数（hankel 函数）	erfinv	逆误差函数
besseli	改进的第一类 bessel 函数	expint	指数积分函数
besselk	改进的第二类 bessel 函数	gamma	gamma 函数
besselj	第一类 bessel 函数	gammainc	非完全 gamma 函数
bessely	第二类 bessel 函数	gammaln	gamma 对数函数
beta	beta 函数	gcd	最大公约数
betainc	非完全的 beta 函数	lcm	最小公倍数
betaln	beta 对数函数	log2	分割浮点数
ellipj	Jacobi 椭圆函数	legendre	legendre 伴随函数
ellipke	完全椭圆积分	pow2	基 2 标量浮点数
erf	误差函数	rat	有理逼近
erfc	互补误差函数	rats	有理输出

附录 7　坐 标 转 换

函　数　名	功能描述	函　数　名	功能描述
cart2pol	笛卡尔坐标到极坐标转换	pol2cart	极坐标到笛卡尔坐标转换
cart2sph	笛卡尔坐标到球面坐标转换	sph2cart	球面坐标到笛卡尔坐标转换

附录 8　矩 阵 函 数

附录 8.1　矩阵分析

函　数　名	功能描述	函　数　名	功能描述
cond	求矩阵的条件数	rcond	LINPACK 倒数条件估计
det	求矩阵的行列式	rref	矩阵的行阶梯型实现
norm	求矩阵的范数	rrefmovie	消元法解方程演示
null	右零空间	subspace	子空间
orth	正交空间	trace	求矩阵的迹
rank	求矩阵的秩		

附录 8.2　线性方程

函　数　名	功能描述	函　数　名	功能描述
/,\	线性方程求解	nnls	非零最小二乘
chol	Cholesky 分解	pinv	求伪逆矩阵
inv	矩阵求逆	qr	矩阵的 QR 分解
lscov	最小二乘方差	qrdelete	QR 分解中删除一行
lu	矩阵的 LU 三角分解	qrinsert	QR 分解中插入一行

附录 8.3　特征值与奇异值

函　数　名	功 能 描 述	函　数　名	功 能 描 述
banlance	改进特征值精度的均衡变换	qz	QZ 算法求矩阵特征值
cdf2rdf	复块对角阵到实块对角阵转换	rsf2csf	实块对角阵到复块对角阵转换
eig	求矩阵的特征值和特征向量	schur	Schur 分解
hess	求 Hessenberg 矩阵	svd	奇异值分解
poly	求矩阵的特征多项式		

附录 8.4　矩阵函数

函　数　名	功 能 描 述	函　数　名	功 能 描 述
expm	矩阵指数函数	logm	矩阵对数函数
funm	矩阵任意函数	sqrtm	矩阵平方根

附录 9　数据分析与 Fourier 变换函数

附录 9.1　基本运算

函　数　名	功 能 描 述	函　数　名	功 能 描 述
cumprod	向量累积	prod	对向量中各元素求积
cumsum	向量累加	sort	对向量中各元素排序
max	求向量中最大元素	sortrows	对矩阵中各行排序
min	求向量中最小元素	std	求向量中各元素标准差
mean	求向量中各元素均值	sum	对向量中各元素求和
median	求向量中中间元素	trapz	梯形法求数值积分

附录 9.2　微分计算

函　数　名	功 能 描 述	函　数　名	功 能 描 述
del2	离散 Laplace 变换	gradient	梯度计算
diff	差分于近视微分		

附录 9.3　滤波与卷积

函　数　名	功 能 描 述	函　数　名	功 能 描 述
Conv	卷积与多项式乘法	filter	一维数字滤波
conv2	二维卷积	filter2	二维数字滤波
Deconv	因式分解与多项式乘法		

附录 9.4　方差处理

函　数　名	功 能 描 述	函　数　名	功 能 描 述
corrcoef	相关系数计算	cov	协方差计算

附录 9.5　Fourier 变换

函　数　名	功 能 描 述	函　数　名	功 能 描 述
abs	绝对值函数	fftshift	fft 与 fft2 输出重排
angle	相角函数	ifft	离散 Fourier 逆变换
cplxpair	依共轭复数对重新排序	ifft2	二维离散 Fourier 逆变换
fft	离散 Fourier 变换	unwrap	相角矫正
fft2	二维离散 Fourier 变换		

附录 10　多项式处理函数

附录 10.1　多项式处理

函　数　名	功 能 描 述	函　数　名	功 能 描 述
conv	卷积与多项式乘法	polyfit	数据的多项式拟合
deconv	因式分解与多项式乘法	polyval	多项式求值
poly	求矩阵的特征多项式	polyvalm	多项式矩阵求值
polyder	多项式求导	residue	部分分式展开
polyeig	多项式特征值	roots	求多项式的根

附录 10.2　数据插值

函　数　名	功 能 描 述	函　数　名	功 能 描 述
griddata	数据网格的插值生成	interpft	一维插值(FFT 方法)
interp1	一维插值(查表)	interpn	多维插值(查表)
interp2	二维插值(查表)	meshgrid	构造三维图形用 x,y 阵列
interp3	三维插值(查表)	spline	三次样条插值

附录 11　非线性数值方法

函　数　名	功 能 描 述	函　数　名	功 能 描 述
dblquad	双重积分	odeget	获得微分方程求解的可选参数
fmin	单变量最优化函数	odeset	设置微分方程求解的可选参数
fmins	多变量最优化函数	quad	低阶数值积分方法
ode45,ode23,odel13,ode15sode23s	微分方程数值解法	quad8	高阶数值积分方法
odefile	对文件定义的微分方程求解		

附录 12 稀疏矩阵函数

附录 12.1 基本稀疏矩阵

函 数 名	功 能 描 述	函 数 名	功 能 描 述
spdiags	稀疏对角矩阵	sprandn	稀疏正态分布随机矩阵
speye	稀疏单位矩阵	sprandsym	稀疏对称随机矩阵
sprand	稀疏均匀分布随机矩阵		

附录 12.2 稀疏矩阵转换

函 数 名	功 能 描 述	函 数 名	功 能 描 述
find	查找非零元素下标	sparse	常规矩阵转换为稀疏矩阵
full	稀疏矩阵转换为常规矩阵	spconvert	由外部格式引入稀疏矩阵

附录 12.3 处理非零元素

函 数 名	功 能 描 述	函 数 名	功 能 描 述
issparse	判断元素是否为稀疏矩阵	spalloc	为非零元素定位存储空间
nnz	稀疏矩阵的非零元素个数	spfun	为非零元素定义处理函数
nonzeros	稀疏矩阵的非零元素	spones	将零元素替换为 1
nzmax	允许的非零元素空间		

附录 12.4 稀疏矩阵可视化

函 数 名	功 能 描 述	函 数 名	功 能 描 述
gplot	绘制图论图形	spy	绘制稀疏矩阵结构

附录 12.5 排序算法

函 数 名	功 能 描 述	函 数 名	功 能 描 述
colmmd	列最小度排序	randperm	产生随机置换向量
colperm	由非零元素的个数来排序各列	symmd	对称最小度排序
dmperm	Dulmage-Mendelsohn 分解	symrcm	反向 Cuthill-McKee 排序

附录 12.6 范数、条件数

函 数 名	功 能 描 述	函 数 名	功 能 描 述
condest	估算$\|\|*\|\|_1$范数	normest	估算$\|\|*\|\|_2$范数
sprank	计算结构秩		

附录 12.7　特征值与奇异值

函　数　名	功能描述	函　数　名	功能描述
eigs	求稀疏矩阵特征值和特征向量	svds	稀疏矩阵奇异值分解

附录 12.8　其他

函　数　名	功能描述	函　数　名	功能描述
spaugment	最小二乘算法形成	symbact	符号因子分解
spparms	设置稀疏矩阵参数		

附录 13　图 形 绘 制

附录 13.1　基本二维图形

函　数　名	功能描述	函　数　名	功能描述
fill	填充二维多边形	polar	极坐标图形绘制
loglog	全对数二维坐标绘制	semilogx	x 轴半对数坐标图形绘制
plot	线性坐标图形绘制	semilogy	y 轴半对数坐标图形绘制

附录 13.2　基本三维图形绘制

函　数　名	功能描述	函　数　名	功能描述
fill3	三维多边形填充	plot3	三维线或点型图绘制
mesh	三维网格图形绘制	surf	三维表面图形绘制

附录 13.3　三维颜色控制

函　数　名	功能描述	函　数　名	功能描述
brighten	图形亮度调整	hidden	网格图的网格线开关设置
caxis	坐标轴伪彩色设置	shading	设置渲染模式
colormap	调色板设置		

附录 13.4　三维光照模型

函　数　名	功能描述	函　数　名	功能描述
diffuse	图像漫射处理	surfl	带光照的三维表面绘制
lighting	光照模式设置	surfnorm	曲面法线
specular	设置镜面反射		

附录 13.5 标准调色板设置

函 数 名	功 能 描 述	函 数 名	功 能 描 述
bone	带有蓝色调的灰度的调色板	hot	以黑红黄白为基色的调色板
cool	以天蓝粉色为基色的调色板	hsv	色度饱和度亮度调色板
copper	线性铜色调的调色板	pink	粉色色调的调色板
flag	以红白蓝黑为基色的调色板	prism	光谱颜色表
gray	线性灰度调色板		

附录 13.6 三维视点控制

函 数 名	功 能 描 述	函 数 名	功 能 描 述
rotate3d	设置三维旋转开关	viewmtx	求视转换矩阵
view	设置视点		

附录 13.7 坐标轴控制

函 数 名	功 能 描 述	函 数 名	功 能 描 述
axis	坐标轴标度设置	hold	设置当前图形保护模式
axes	坐标轴位置设置	subplot	将图形窗口分成几个区域
box	坐标轴盒状显示	zoom	二维图形缩放
grid	坐标网格线开关设置		

附录 13.8 图形注解

函 数 名	功 能 描 述	函 数 名	功 能 描 述
colorbar	颜色条设置	xlabel	给图形的 x 轴加文字说明
gtext	在鼠标位置加文字说明	ylabel	给图形的 y 轴加文字说明
text	在图形上加文字说明	zlabel	给图形的 z 轴加文字说明
title	给图形加标题		

附录 13.9 拷贝与打印

函 数 名	功 能 描 述	函 数 名	功 能 描 述
print	打印图形或将图形存盘	orient	设置纸的方向
printopt	设置打印机为默认值		

附录 14　特 殊 图 形

附录 14.1　特殊二维图形

函　数　名	功 能 描 述	函　数　名	功 能 描 述
area	区域填充	feather	羽状图形绘制
bar	条形图绘制	fplot	给定函数绘制
barh	水平条形图绘制	hist	直方图绘制
bar3	3 维条形图绘制	pareto	pareto 图绘制
bar3h	3 维水平条形图绘制	pie	饼状图绘制
comet	彗星状轨迹绘制	stem	离散序列图形绘制
errorbar	误差条形图绘制	stairs	梯形图绘制

附录 14.2　等高线及其他二维图形

函　数　名	功 能 描 述	函　数　名	功 能 描 述
contour	等高线绘制	pcolor	伪色绘制
contourf	等高线填充绘制	quiver	有向图(箭头)绘制
contour3	三维等高线绘制	voronoi	voronoi 图绘制
clabel	等高线高程标志		

附录 14.3　特殊三维图形

函　数　名	功 能 描 述	函　数　名	功 能 描 述
comet3	三维彗星状轨迹绘制	slice	切片图
meshc	带等高线的三维网格绘制	surfc	带等高线的三维表面绘制
meshz	带零平面的三维网格绘制	trisurf	表面图形的三角绘制
stem2	杆图绘制	trimesh	网格图形的三角绘制
quiver3	三维箭头(有向图)绘制	waterfall	瀑布型图形绘制

附录 14.4　图像显示与文件 I/O

函　数　名	功 能 描 述	函　数　名	功 能 描 述
brighten	图形色调亮化	image	图像显示
colorbar	颜色条设置	imfinfo	图形文件信息
colormap	调色板设置	imread	从文件读取图像
contrast	灰度对比度设置	imwrite	保存图像

附录 14.5　动画处理

函　数　名	功 能 描 述	函　数　名	功 能 描 述
capture	屏幕抓取	movie	播放动画帧
getframe	获取动画帧		

附录 14.6　实体模型

函　数　名	功 能 描 述	函　数　名	功 能 描 述
cylinder	圆柱体生成	sphere	球体生成

附录 15　图 形 处 理

附录 15.1　图形窗口生成与控制

函　数　名	功 能 描 述	函　数　名	功 能 描 述
clf	清除当前图形窗口	gcf	获得当前图形的窗口句柄
close	关闭图形窗口	refresh	图形窗口刷新
figure	生成图形窗口	shg	显示图形窗口

附录 15.2　坐标轴建立与控制

函　数　名	功 能 描 述	函　数　名	功 能 描 述
axes	坐标轴标度设置	gca	获得当前坐标轴句柄
axis	坐标轴位置设置	hold	设置当前图形保护模式
box	坐标轴盒状显示	ishold	返回 hold 的状态
caxis	为彩色坐标轴刻度	subplot	将图形窗口分成几个区域
cla	清除当前坐标轴		

附录 15.3　处理图形对象

函　数　名	功 能 描 述	函　数　名	功 能 描 述
axes	坐标轴生成	surface	表面生成
figure	图形窗口生成	text	文本生成
image	图像生成	unicontrol	生成一个用户接口控制
light	光源生成	uimenu	菜单生成
line	线生成		

附录 15.4　图形

函　数　名	功 能 描 述	函　数　名	功 能 描 述
copyobj	图形对象拷贝	gcbo	获得当前回调对象的句柄
delete	对象删除	gco	获得当前对象的句柄
drawnow	清除未决的图形对象事件	get	获得对象属性
findobj	查找对象	reset	重新设置对象属性
gebf	获得当前回调窗口的句柄	set	设置对象属性

附录 16　GUI(图形用户接口)

附录 16.1　GUI 函数

函　数　名	功 能 描 述	函　数　名	功 能 描 述
ginput	获取鼠标输入	uiresume	继续执行
selectmove-resize	对象的选择、移动、大小设置、拷贝	uiwait	中断执行
uicontrol	生成图形用户接口对象	waitforbut-terpress	等待按钮输入
uimenu	生成菜单对象	waitfor	中断执行

附录 16.2　GUI 设计工具

函　数　名	功 能 描 述	函　数　名	功 能 描 述
align	坐标轴与用户接口控制的对齐工具	menuedit	菜单编辑器
cbedit	回调函数编辑器	propedit	属性编辑器
guide	GUI 设计工具		

附录 16.3　对话框

函　数　名	功 能 描 述	函　数　名	功 能 描 述
dialog	对话框生成	printdlg	打印对话框
axlimdlg	坐标轴设限对话框	questdlg	请求对话框
errordlg	错误对话框	uigetfile	标准的打开文件对话框
helpdlg	帮助对话框	uiputfile	标准的保存文件对话框
inputdlg	输入对话框	uisetcolor	颜色选择对话框
listdlg	列表选择对话框	uisetfont	字体选择对话框
msgdlg	消息对话框	waitbar	等待条显示
pagedlg	页位置对话框	warndlg	警告对话框

附录 16.4　菜单

函　数　名	功 能 描 述	函　数　名	功 能 描 述
makemenu	生成菜单结构	umtoggle	菜单对象选中状态切换
menubar	设置菜单条属性	wimenu	生成 window 菜单项的子菜单

附录 16.5　组按钮

函　数　名	功 能 描 述	函　数　名	功 能 描 述
btndown	组按钮中的按钮按下	btnstate	查询组按钮中的按钮状态
btngroup	组按钮生成	btnup	组按钮中的按钮弹起
btnpress	组按钮中的按钮按下管理		

附录 16.6　自定义窗口属性

函　数　名	功能描述	函　数　名	功能描述
clruprop	清除用户自定义属性	setuprop	设置用户自定义属性
getuprop	获取用户自定义属性		

附录 16.7　其他应用

函　数　名	功能描述	函　数　名	功能描述
allchild	获取所有子对象	popupstr	获取弹出式菜单选中项的字符串
edtext	坐标轴文本对象编辑	remapfig	改变窗口中对象的位置
findall	查找所有对象	setptr	设置窗口指针
getptr	获得窗口指针	setstatus	设置窗口中文本串状态
getstatus	获取窗口中文本串状态		

附录 17　声 音 处 理

函　数　名	功能描述	函　数　名	功能描述
sound	将向量转换成声音	wavread	读.wav 文件
auread	读.au 文件	wavwrite	写.wav 文件
auwrite	写.au 文件		

附录 18　字符串处理函数

附录 18.1　字符串处理

函　数　名	功能描述	函　数　名	功能描述
strings	Matlab 字符串函数说明	upper	字符串大写
isstr	字符串判断	lower	字符串小写
deblank	删除结尾空格	isletter	字母判断
str2mat	字符串转换成文本	isspace	空字符判断
strcmp	字符串比较	strrep	子串替换
findstr	子串查找	strtok	标记查找

附录 18.2　字符串与数值转换

函　数　名	功能描述	函　数　名	功能描述
num2str	变数值为字符串	sprintf	数值的格式输出
str2num	变字符串为数值	sscanf	数值的格式输入
int2str	变整数为字符串		

附录 18.3　进制转换

函　数　名	功 能 描 述	函　数　名	功 能 描 述
hex2num	十六进制到 IEEE 标准下浮点数的转换	hex2dec	十六进制到十进制的转换
dec2hex	十进制到十六进制的转换		

附录 19　文件输入输出函数

附录 19.1　基本文件输入输出

函　数　名	功 能 描 述	函　数　名	功 能 描 述
fclose	关闭文件	feof	文件结尾检测
fopen	打开文件	ferror	文件 I/O 错误查询
fread	读二进制流文件	frewind	文件指针回绕
fwrite	写二进制流文件	fseek	设置文件指针位置
fgetl	读文本文件(无行结束符)	ftell	获得文件指针位置
fgets	读文本文件(含行结束符)	sprintf	格式化数据转换为字符串
fprintf	写格式化数据到文件	sscanf	依数据格式读取字符串
fscanf	从文件读格式化数据		

附录 19.2　特殊文件输入输出

函　数　名	功 能 描 述	函　数　名	功 能 描 述
imfinfo	获得图形文件信息	wklread	读一 Lotus 123 WK1 数据表
imread	图像的文件读取	wklwrite	将一矩阵写入 Lotus 123 WK1 数据表文件
imwrite	图像的文件保存	xlgetrange	读 Excel 表格文件的数据
qrwrite	保存一段 QuickTime 电影文件	xlsetrange	写 Excel 文件

附录 20　位　操　作

函　数　名	功 能 描 述	函　数　名	功 能 描 述
bitand	位求与	bitor	位求或
bitcmp	位求补	bitset	位设置
bitget	位获取	bitshift	位移动
bitmax	求最大无符号浮点整数	bitxor	位异或

附录 21 复杂数据类型

附录 21.1 数据类型

函 数 名	功 能 描 述	函 数 名	功 能 描 述
cell	生成单元数组	sparse	生成稀疏数组
char	生成字符串	struct	生成结构
double	转换成双精度型	uint8	转换成无符号单字节整数
inline	生成 INLINE 对象		

附录 21.2 结构操作

函 数 名	功 能 描 述	函 数 名	功 能 描 述
fieldnames	获得结构的字段名	rmfield	删除结构字段
getfield	获得结构的字段值	setfield	设置结构的字段值
isfield	如果字段属于结构则返回真	struct	生成结构数组
isstruct	如果是结构则返回真	struct2cell	结构到单元数组的转换

附录 21.3 多维数组操作

函 数 名	功 能 描 述	函 数 名	功 能 描 述
cat	数组连接	permute	任意改变矩阵维数序列
ipermute	任意改变矩阵维数序列	shiftdim	矩阵维数序列的左移变换
ndims	求矩阵维数	squeeze	去除多维数组中的一维向量
ndgrid	N 维数组生成		

附录 21.4 单元数组操作

函 数 名	功 能 描 述	函 数 名	功 能 描 述
cell	单元数组生成	iscell	如果是单元数组则返回真
celldisp	显示单元数组内容	num2cell	将数值数组转换为单元数组
cellplot	单元数组内容的图形显示	struct2cell	将结构数组转换为单元数组
cell2struct	单元数组转换成结构数组		

附录 21.5 面向对象函数

函 数 名	功 能 描 述	函 数 名	功 能 描 述
class	生成一个类对象	methods	显示所有方法名
isa	如果是某一给定类对象则返回真	struct	将对象转换为结构数组
isobject	如果是一个对象则返回真	superiorto	建立类间的关系
inferiorto	建立类间的关系		

附录 22　日期与时间

函　数　名	功 能 描 述	函　数　名	功 能 描 述
now	以数字形式给出当前日期和时间	weekday	星期函数
date	以字符串形式给出当前日期	eomday	月末日判断函数
clock	以向量形式给出当前日期和时间	cputime	所用的 CPU 时间
datenum	日期的数字形式转换	tic	启动秒表计时器
datestr	日期的字符串形式转换	toc	读取秒表计时器
datevec	日期的向量形式转换	etime	使用时间函数
calendar	日历函数	pause	暂停函数

附录 23　动态数据交换

函　数　名	功 能 描 述	函　数　名	功 能 描 述
ddeadv	设置 DDE 连接	ddereq	接收数据
ddeexec	发送要执行的串	ddeterm	DDE 终止
ddeinit	DDE 初始化	ddeunadv	释放 DDE 连接
ddepoke	发送数据		

参 考 文 献

[1]　Using MATLAB, MathWorks, 1997

[2]　MATLAB Function Reference, MathWorks, 1997